Tactical Sugarcane Harvest Scheduling

Björn Jonas Stray

Dissertation presented for the degree of
Doctor of Philosophy (Operations Research)
at Stellenbosch University

Promoter: Prof JH van Vuuren
Co-promoter: Prof CN Bezuidenhout

December 2010

Declaration

By submitting this dissertation electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the owner of the copyright thereof (unless to the extent explicitly otherwise stated) and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

December 8, 2010

Abstract

Computerised sugarcane harvest scheduling decision support is an active field of research which ties in closely with the broader problem of automating and streamlining the various activities in the sugar supply chain. In this dissertation, the problem of providing decision support with respect to sugarcane harvesting decisions is defined within a number of contexts, each representing a typical kind of organisation of sugarcane farmers into a cohesive decision making unit with its specific requirements and limitations that exist in practice. A number of variations relevant to these contexts of an overarching *tactical sugarcane harvest scheduling problem* (THSP) are considered and solved in this dissertation. The THSP is the problem of *providing objective, responsible decision support to persons charged with the task of determining optimal harvesting dates for a set of sugarcane fields across an entire season.*

Sugarcane fields typically differ in terms of the age, variety, life-cycle stage and in many other properties of the cane grown on them. The growth of sugarcane crops may also be affected by environmental conditions such as accidental fires, frosts or storms which have a detrimental effect on crop-value. Since sugarcane is a living organism, its properties change over time, an so does the potential profit associated with it. The practicalities of farming cause further complication of the problem (for example, seasonal changes alter the conditions under which the crop is harvested and transported). The rainy season carries with it the added cost of disallowing long-range vehicles to drive into the fields, forcing the unloading and reloading of cane at so-called *loading zones.* Other considerations, such as the early ploughing out of fields to allow them to fallow before being replanted, compounds the THSP into a multi-faceted difficult problem requiring efficient data management, mathematical modelling expertise and efficient computational work.

In the literature the THSP has been viewed from many different standpoints and within many contexts, and a variety of operations research methodologies have been employed in solving the problem in part. There is, however, no description in the literature of a solution to the THSP that takes the negative effects of extreme environmental conditions on the quality of a harvesting schedule into account in a scientifically justifiable manner; most models in the literature are based on optimising sucrose yield alone under normal conditions, rendering weak schedules in practice. The scope of the modelling and solution methodologies employed in this dissertation towards solving the THSP is restricted to integer programming formulations and approximate solution methods. The parameters associated with these models were determined empirically using historical data, as well as previous work on deterioration of sugarcane following environmental and other events.

The THSP is solved in this dissertation by designing a generic architecture for a conceptual *decision support system* (DSS) for the THSP in the various contexts referred to above, which is capable of accommodating the effects of extra-ordinary environmental conditions, as well as the introduction of a computer-implemented version of a real DSS for the THSP conforming to the framework of this generic architecture. The DSS building blocks include prediction models for sugarcane yield, sugarcane recoverable value under normal circumstances, the costs associated with a harvesting schedule and the negative effects on sugarcane recoverable value of extraordinary environmental conditions. The working of the DSS is based on a combinatorial optimisation model resembling the well-known *asymmetric travelling salesman problem with time-dependent costs* which is solved approximately by means of an attribute-based tabu search in which both local and global moves have been incorporated. The DSS is also validated by experienced sugarcane industry experts in terms of the practicality and quality of the schedules that it produces.

Uittreksel

Gerekenariseerde besluitsteun vir die skedulering van suikerriet-oeste is 'n aktiewe navorsingsveld wat nou verwant is aan die breër probleem van die outomatisering en vaartbelyning van 'n verskeidenheid aktiwiteite in die suikervoorsieningsketting. Die probleem van die daarstelling van steun rakende suikkerriet oestingsbesluite word in hierdie proefskrif in 'n aantal kontekste oorweeg, elk met betrekking tot 'n tipiese soort organisasie van suikerrietboere in 'n samehorige besluitnemingseenheid met sy spesifieke vereistes en beperkings in die praktyk. Verskeie variasies van 'n oorkoepelende *taktiese suikerriet-oesskeduleringsprobleem* (TSOSP) wat in hierde konteksste relevant is, naamlik die probleem om objektiewe, verantwoordbare steun aan besluitnemers te bied wat verantwoordelik is vir die bepaling van optimale oesdatums vir 'n versameling suikerrietplantasies oor die bestek van 'n hele seisoen, word in hierdie proefskrif bestudeer en opgelos.

Suikerrietplantasies verskil tipies in terme van ouderdom, gewastipe, posisie in die lewensiklus, en vele ander eienskappe van die suikerriet wat daar groei. Omgewingstoestande, soos onbeplande brande, ryp of storms, het verder ook 'n negatiewe impak op die waarde van suikerriet op sulke plantasies. Omdat suikerriet 'n lewende organisme is, verander die eienskappe daarvan oor tyd, en so ook die potensiële wins wat daarmee geassosieer word. Boerderypraktyke bemoeilik verder die skeduleringsprobleem onder beskouing (seisoenale veranderings beïnvloed byvoorbeeld die wyse waarop suikerriet ge-oes en vervoer word). Addisionele koste gaan voorts met die reënseisoen gepaard, omdat die plantasies dan nie toeganklik is vir langafstand transportvoertuie nie en suikerriet gevolglik na spesiale laaisones gekarwei moet word voordat dit op hierdie voertuie gelaai kan word. Ander oorwegings, soos die vroeë uitploeg van plantasies sodat die grond kan rus voordat nuwe suikerriet aangeplant word, veroorsaak dat die TSOSP 'n moeilike multi-faset probleem is, wat goeie databestuur, wiskundige modelleringsvernuf en doeltreffende rekenaarwerk vereis.

Die TSOSP word in die literatuur vanuit verskillende standpunte en in verskeie kontekste oorweeg, en 'n aantal uiteenlopende operasionele navorsingsmetodologieë is al ingespan om hierdie probleem ten dele op te los. Daar is egter geen poging in die literatuur om 'n oplossing vir die TSOSP daar te stel waarin daar op 'n wetenskaplik-verantwoordbare wyse voorsiening gemaak word vir die negatiewe effekte wat uitsonderlike omgewingstoestande op die kwaliteit van oesskedules het nie; die meeste modelle in die literatuure is op slegs sukrose-opbrengs onder normale omstandighede gebaseer, wat lei na swak skedules in die praktyk. Die bestek van die wiskundige modellerings- en gepaardgaande oplossings-metodologieë word in hierdie proefskrif vir die TSOSP beperk tot onderskeidelik heeltallige programmeringsformulerings en die bepaling van benaderde oplossings deur lokale soekprosedures. Die parameters wat met hierdie modelle en soekmetodes geassosieer word, word empiries bepaal deur gebruikmaking van historiese data asook bestaande werk oor die degradering van suikerriet as gevolg van omgewings- en ander eksterne faktore.

Die TSOSP word in hierdie proefskrif opgelos deur die ontwerp van 'n generiese argitektuur vir 'n konseptuele *besluitsteunstelsel* (BSS) vir die TSOSP in die onderskeie kontekste waarna hierbo verwys word en wat die effekte van uitsonderlike omgewingsfaktore in ag neem, asook die daarstelling van 'n rekenaar-geïmplementeerde weergawe van 'n daadwerklike BSS vir die TSOSP wat in die raamwerk van hierdie generiese argitektuur pas. Die boustene van hierdie BSS sluit modelle in vir die voorspelling van suikerrietopbrengs, die herwinbare waarde van suikerriet onder normale omstandighede, die verwagte koste geassosieer met 'n oesskedule en die negatiewe effekte van omgewingsfaktore op die herwinbare waarde van suikerriet. Die werking van die BSS is gebaseer op 'n kombinatoriese optimeringsprobleem wat aan die welbekende *asimmetriese handelreisigersprobleem met tyd-afhanklike kostes* herinner, en hierdie model word benaderd opgelos deur middel van 'n eienskap-gebaseerde tabu-soektog waarin beide lokale en globale skuiwe geïnkorporeer is. Die BSS word ook gevalideer in terme van die haalbaarheid en kwaliteit van die skedules wat dit oplewer, soos geassesseer deur ervare kundiges in die suikerrietbedryf.

Terms of reference

The idea of automated scheduling of sugarcane harvesting operations was introduced to the author in 2007 by Professor Carel Bezuidenhout of the School of Bioresources Engineering and Environmental Hydrology (BEEH) at the University of KwaZulu-Natal (UKZN). Professor Bezuidenhout was on sabbatical in the United Kingdom at the time, visiting the Cranfield School of Management, Cranfield University and there gave a presentation on current supply chain problems in the South African sugarcane industry. Dr Roy Andersson of the Industrial Engineering Department, School of Engineering, University of Borås attended the presentation, noticed that some of the problems within the South African sugarcane supply chain may be approached by means of operations research tools, and was aware that the author was highly interested in pursuing a doctoral degree in operations research. The author invited Professor Bezuidenhout to give the presentation in Sweden, at the Industrial Engineering Department of the University of Borås where the author worked. The School of Engineering at the University of Borås subsequently decided to provide funding in order to enable the author to work on a doctoral dissertation project and Professor Bezuidenhout advised the author to apply for admission to the doctoral programme in operations research at Stellenbosch University, under the supervision of Professor Jan van Vuuren, head of the Operations Research Division within the Department of Logistics at Stellenbosch University, and under the co-supervision of Professor Bezuidenhout. Professor Bezuidenhout advised towards applying to the Division of Operations Research due to the complementary component it would bring to the doctoral dissertation project, in terms of a well-developed research environment in the subject of operations research with prior experience in modelling problems within the sugarcane industry. The author commenced his PhD studies on 21 January 2008.

The research group at the Operations Research Division of Stellenbosch University is involved in projects spanning a multitude of practical applications, and there are currently two students working on problems involving sugarcane at the division. Ms Linke Potgieter is currently working on a masters project with the title "*A mathematical model for the control of the stalk borer* Eldana saccharina *Walker*" focussing on a method called the *sterile insect technique*. This technique is based on the idea that sterile specimens of the insect are introduced into the insect population with the purpose of diluting the presence of fertile specimens, thereby decreasing the successful fertilisations and thus the insect population size. Ms Potgieter is working on a part of a larger project including researchers with the South African Sugarcane Research Institute (SASRI). Ms Heletje van Staden—Namibia's top female cyclist—is currently working on a BSc Honours year project within the Operations Research Division with the title "*Resource assignment and scheduling in sugarcane industry*" involving sugarcane resources—especially with respect to irrigation—scheduling. Both Ms Potgieter and Ms van Staden are supervised by Professor van Vuuren.

Parts of this project were conducted *in situ* and thus the author was stationed in Pietermaritzburg, KwaZulu-Natal for much of its duration. The School of BEEH at the Pietermaritzburg campus of UKZN continuously conducts a wide range of projects involving various aspects of sugarcane production. Professor Bezuidenhout is a SASRI Senior Research Fellow at the School of BEEH and is responsible for many of these projects housed under the subject area of agricultural engineering. Some current projects include a masters project on *Diagnostics of integrated sugarcane production systems* by Milindi Sibomana, a doctoral project on the *Improvement of sugar cane supply and processing chain using complex systems thinking tools* by Thawani Sanjika, and a doctoral project on *Optimising the raw material transportation of the sugar value chain* by Louis Lagrange.

The idea to attempt to provide decision support in sugarcane harvesting operations scheduling was conceived some time into examining the reasons behind a problem which was thought to be a *crisis management* problem arising across entire mill areas in South Africa when sugarcane is struck by destructive environmental events. The crisis perspective remained as the main issue until it was proposed that *tactical harvest scheduling* may be the solution to crisis-type problems as well, not just the solution to the problem of maximising seasonal profits. This proposal came as a result of the research conducted while the author was stationed in Pietermaritzburg, close to several sugar mills in KwaZulu-Natal.

While stationed in Pietermaritzburg, the author visited Noodsberg Mill on several occasions, interviewing the cane supply manager, as well as taking a tour of the mill. The author participated in several meetings organised by the Noodsberg Mill Group Board between representatives of Noodsberg Canegrowers, the School of BEEH (including Professor Bezuidenhout and various students that he supervised), the Noodsberg Mill Area Extension Officers (including Mr Pat Brenchley), the Noodsberg Mill (including Mr Julius De Lange) as well as various individually acting cane growers. The author also attended a proper Mill Group Board meeting in Noodsberg. Furthermore, Professor Bezuidenhout and the author met with the cane supply manager at Sezela Mill, Mr Allan Simpson, who over several hours rendered the insight of a long-time industry professional into the problem of harvest scheduling from a miller's perspective to the author.

At the time of these meetings, the problem was still being formulated, and the author hence conducted a number of question-driven interviews with several growers. During these interviews, the grower was encouraged to elaborate on his/her answer following a predetermined question and the author would subsequently ask *ad hoc* follow-up questions in response to the grower's answer. This was done in order to understand the harvesting problem from the growers' perspectives. It was during these interviews that the problem was understood to a sufficient extent in order for the author to be able to attempt a general problem formulation, as well as to select an appropriate modelling approach.

During a subsequent meeting with Mr Edgar Bruggeman, then Extension Officer at Eston Mill, the idea was conceived to test the problem modelling approach on a syndicated group of growers that he knew personally, in the Eston Mill area. The syndicate members were Mr Roger O'Neill, Mr Clive Coulthard, Mr Eric Lewis and Mr Malcolm Thompson and the syndicate had operated for a few years, employing Mr Shawn Kyle as the manager of the harvesting operation. This group was approached with a proposal including that the author would send suggested harvesting schedules to the syndicate on a regular basis during the 2009 harvesting season and that the growers should return information in response to these suggested schedules concerning the actually harvested fields since the last suggested schedule received as well as their opinions concerning the quality of the schedules. The schedules would be constructed based on data that the growers were to send to the author ahead of the season. The syndicate unanimously agreed, and the author proceeded to spend time with the growers and the manager, in order to better understand the details of how harvesting operations are managed and how harvest scheduling decisions are made in practice. While this learning (and data collection) phase took place, work on implementing the first *decision support system* (DSS) of this dissertation was commenced. The syndicate upheld their part of the agreement throughout the season.

The actual implementation of the first DSS was completed in the postgraduate laboratory operated by Professor Jan van Vuuren at the Operations Research Division, with the aid and advice of additional experienced operations researchers, namely Dr Alewyn Burger and Mr Neil Jacobs. This DSS was ready to produce schedules ahead of the 2009 harvesting season and did so regularly until September 2009, when it was decided to stop the schedule generation procedure due to the small number of fields remaining. The author again spent time with the growers and gathered comments that had not been recorded from the regular communication (exchanged in conjunction with the submission of the schedules) during the 2009 harvesting season. He then implemented a second version of the DSS. This final version of the DSS was to incorporate aspects which were not incorporated into the first DSS, and was to be tested during the 2010 harvesting season by the same syndicate, which had been agreed upon during the syndicate's end-of-season meeting in November 2009.

The testing of the final DSS occurred while the author was stationed at the Operations Research Division of Stellenbosch University, enabling the employment of the computational resources and communication facilities required for a successful experiment and access to the skills possessed by the operations researchers within the division to ensure the application of appropriate changes to the DSS architecture and computer implementation.

Acknowledgements

The author hereby wishes to express his sincere and humble gratitude towards Professor Jan van Vuuren for his unwavering faith in the author's abilities, for his steadfast guidance of the work, for his respectable ability to understand the details of all things that the author wrote and Professor Van Vuuren read, for allowing the author to participate in the operations research conferences of the Operations Research Society of South Africa and the occasional laughs, wine-tastings and good times.

Professor Carel Bezuidenhout is thanked with deep gratitude for the invaluable insight into the problem, his free sharing of his immense network in the sugar industry, for allowing the author to participate in meetings within his highly relevant projects and also for his care and consideration in aiding the author in finding his footing on the South African soil.

The author wishes to thank his wife, who is the anchor and love of his life. She has been the greatest encourager and lifter during hard and heavy moments and she has been the sharer of joy in times of strengths and success. She has supported the author in the best manner there is, which is in complete selflessness.

Roger O'Neill, who sadly died of cancer early in 2010, is hereby thanked in the deepest sense of the word, and the greatest of sympathy is extended to his family. Roger opened his wealth of knowledge in sugarcane farming to the author, as well as opened his home, and was second to none in running a flawless farming operation and in being a friend.

Shawn Kyle is thanked for his invaluable work during the entire validation effort and for his honest comments and advice. Shawn is also thanked for opening his home and being very patient and accommodating towards the—especially in the beginning—not so practically knowledgeable author.

Eric Lewis, Clive Coulthard and Malcolm Thompson are thanked for freely sharing their thoughts on the problem, their knowledge about sugarcane farming, their time and their historical field records with the author. Eric, Clive and Malcolm are deeply thanked for inviting the author to their homes, especially to the end-of-the-season braai.

The author extends his gratitude towards Julius de Lange of Noodsberg Mill, who on numerous occasions welcomed the author to partake in meetings and in industrial data-sets as well as by opening his vast network of contacts within the industry.

Edgar Bruggeman contributed to this work by sharing his thoughts on the problem, as well as being responsible for connecting the author with the growers mentioned above, making it possible to perform the validation experiments in a fruitful environment, and for this he is hereby acknowledged.

Professor Peter Lyne is thanked for his valuable insight and sound advice in formulating the early models, for his practical help in settling the author in the country and for being generally interested in the project.

Rear Admiral (ret.) Dr Gert Engelbrecht is thanked for inviting the author to see and experience the Kalahari as well as to the meetings that the admiral held in the sugarcane supply chain project he manages for CSIR. The meetings provided valuable insights into the present issues in the South African sugar supply chain.

The author wishes to thank the numerous people to whom he spoke informally or who were interviewed by him regarding the harvesting problem, who are not named here.

The Department of Logistics at Stellenbosch University is hereby thanked for the use of their computing facilities and office space. The School of Bioresources Engineering and Environmental Hydrology of the University of KwaZulu-Natal is hereby thanked for the use of their computing facilities and office space. The School of Engineering of the University of Borås is hereby acknowledged for funding the author throughout the three years of the project. The financial support of the South African National Research Foundation (NRF), in the form of a Bursary (under grant number GUN 2072999), and of Stellenbosch University in the form of three Merit Bursaries towards this research are hereby acknowledged. Any opinions or findings in this dissertation are those of the author and do not necessarily reflect the views of Stellenbosch University, University of Borås, University of KwaZulu-Natal or the NRF.

Pectora roborant cultus recti

UNIVERSITEIT • STELLENBOSCH • UNIVERSITY

UNIVERSITY OF BORÅS

SCIENCE FOR THE PROFESSIONS

UNIVERSITY OF
KWAZULU-NATAL

Table of Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

ACO	Ant colony optimisation
ATSP	Asymmetric travelling salesman problem
ATSPTDC	Asymmetric travelling salesman problem with time-dependent costs
ATSPTW	Asymmetric travelling salesman problem with time windows
ANOVA	Analysis of variance
BMF	Base model formulation
CAP	Classical assignment problem
CCS	Commercial cane sugar
CIRAD	Centre de Cooperation Internationale en Recherche Agronomique puor le Développement
CJSP	Classical job-shop scheduling problem
CPU	Central processing unit
DRD	Daily rateable delivery
DMAIC	Design, measure, analyse, improve, control
DSS	Decision support system
DSSDV	Decision support system—development version
EGT	Effective growth time
EU	European Union
GA	Genetic algorithm
GAP	Generalised assignment problem
GATT	General Agreement on Tariffs and Trade
GPS	Global positioning system
GS	Gledhow Sugar Company Limited
HTD	Harvesting time deadline
HTW	Harvesting time window
IB	Inter-nodes bored
Il	Illovo Sugar Limited
MILP	Mixed integer linear programming model
MA	Memetic algorithm
NID	Normally and independently distributed
NP	Non-deterministic polynomial
OR	Operations research
PMF	Performance measurement framework
RMSE	Root mean square error
R&D	Research and development
RSD	Ratoon stunting disease
RUE	Radiation use efficiency
RV	Sugarcane recoverable value

SA	Simulated annealing
SASA	South African Sugar Association
SASRI	South African Sugarcane Research Institute
SB	Stalks bored
SC	Supply chain
SCM	Supply chain management
SD	Stalks damaged
SIA	Sugar industry agreement
SMF	Sequential model formulation
SMSP	Single machine scheduling problem
SS	Scatter search
STDSP	Sequence and time-dependent scheduling problem
VRP	Vehicle routing problem
WTO	World Trade Organisation
TDTSP	Time dependent travelling salesman problem
TH	Tongaat Hulett Sugar
THSP	Tactical sugarcane harvest scheduling problem
TQM	Total quality management
TS	Tabu search
TSL	Transvaal Sugar Limited
TSP	Travelling salesman problem
TSPLP	Travelling salesman problem linear programming relaxation
UCL	Union Cooperative Limited
USM	Umfolozi Sugar Mill Limited

List of Reserved Symbols

Symbols in this dissertation conform to the following font conventions:		
$\mathcal{A}$	Symbol denoting a **problem**	(Calligraphic italic capitals)
A	Symbol denoting a **set**	(Italic capitals)
$\boldsymbol{A}$	Symbol denoting a **matrix**	(Boldface capitals)
$\boldsymbol{a}$	Symbol denoting a **vector**	(Boldface lowercase letters **or** Greek lowercase letters)

Symbol	**Meaning**
Functions & formulas	
$A(e)$	Aspiration function
$a_p(s,\boldsymbol{\omega})$	Attribute function
B_{ij}^M	Base yield model
B_{ij}^R	Base RV model
c_{ij}^d	Zone-loading cost function
c_{ij}^{HTD}	Harvesting time deadline penalty cost function
c_{ij}^{HTW}	Harvesting time window penalty cost function
$\chi_{ik}(\boldsymbol{\omega})$	Indicator function
$\delta_{0,ij\ell_i(k)}^M$	Immediate event effect on cane yield
$\delta_{0,ij\ell_i(k)}^R$	Immediate event effect on RV
$\delta_{1,ij\ell_i(k)}^M$	Daily event effect on cane yield
$\delta_{1,ij\ell_i(k)}^R$	Daily event effect on RV
D	Daily rateable delivery function
E_{ij}^M	Event-driven yield model
E_{ij}^R	Event-driven RV model
$N_{\text{init\_shift}}(\boldsymbol{\sigma})$	BMF local search move function
$N_{\text{perturb}}(\boldsymbol{\sigma})$	BMF local search move function
$N_{\text{shift}}(\boldsymbol{\sigma})$	BMF local search move function
$P(W_{ij})$	Probability of wet conditions on field i during period j
$S(\boldsymbol{\omega})$	All moves of type S possible from the solution $\boldsymbol{\omega}$
$\tau_i(\boldsymbol{\omega})$	Time instant function
$\theta_i(\boldsymbol{\omega})$	Period function
$U_i(\boldsymbol{\omega})$	Index set function that contains fields
V	Objective function in the BMF
z	Objective function in the SMF
Lists	
T^{ad}	Tabu list for added attributes

T^{dr}	Tabu list for deleted attributes
T_p	Tabu list for move reversals
Problems	
$\mathcal{C}$	A combinatorial optimisation problem
$\mathcal{C}'$	A relaxation of $\mathcal{C}$
$\mathcal{S}'_i$	A subproblem of $\mathcal{C}$
Sets	
A	Arc set
A^ϵ	Set of precedence constraints in the CJSP
A^ζ	Arc set in the STDSP
E	Neighbouring pair (attribute) set in the tabu search method
E^β	Edge set in the VRP
E^x	Set of extraneous events
I	Field set
I^α	Agent set in the CAP and the GAP
J	Period set
J^α	Task set in the CAP and the GAP
K	Set of time instants
K^ζ	Set of relevant intervals in the STDSP
N^ϵ	Set of operations in the CJSP
N^ζ	Set of jobs in the STDSP
P^ζ	Set of possible predecessors in the STDSP
S^β	Subtour set in the ATSP and proper subset of customers in the VRP
S^ζ	Set of possible successors in the STDSP
S^η	Subtour set in the TSP
$\boldsymbol{\sigma}$	Encoded solution to the BMF
V	Vertex set
$\mathbb{Z}^+$	All positive integers
$\mathbb{Z}^n_+$	All positive integers in n-space
Ω	Feasible domain of a combinatorial optimisation problem
Decision variables	
$C_{\max}$	Makespan
λ_j	Precipitation rate during period j
μ_{ij}	Dry-out rate on field i during period j
ρ_{ij}	Utilisation factor on field i during period j (wetness)
$w^\delta_{uv\kappa}$	The number of instants by which a tour is early at vertex v
W_{ij}	Binary random variable
x_{ij}	Binary assignment variable
x^α_{ij}	Binary assignment variable
x^β_{uv}	Binary arc variable
$x^\gamma_{uv\alpha}$	Binary arc variable
$x^\delta_{uv\kappa}$	Binary arc variable
x^ζ_{ijk}	Binary arc variable
$\boldsymbol{x}$	Characteristic vector of a solution to a combinatorial optimisation problem

y_u^β	Time instant at which a tour leaves vertex u in the ATSPTW
y_u^δ	Time instant at which a tour leaves vertex u in the ATSPTDC
y_{ij}^ϵ	Time at which job j is processed on machine i
$\boldsymbol{\omega}$	Encoded solution to a combinatorial optimisation problem
Parameters	
$\boldsymbol{A}$	Matrix of coefficients for an integer or linear programming problem
a_u^β	First time instant of a time window in the ATSPTW
a_u^δ	First time instant of a time window in the ATSPTDC
a^ζ	Beginning of forbidden interval in the STDSP
$\boldsymbol{b}$	Vector of coefficients for an integer or linear programming problem
b_i	Resource available for agent i
b_u^β	Last time instant of a time window in the ATSPTW
b_u^δ	Last time instant of a time window in the ATSPTDC
b_ζ	End of forbidden interval in the STDSP
$\hat{\boldsymbol{\beta}}$	Vector of regression coefficients
BRK	Number of breakaways from local optimum during tabu search
c	Coefficient accounting for processing difficulties due to fibre
c_{ij}^α	Cost of assigning agent i to perform task j in the CAP and the GAP
c_{uv}^β	Cost of traversing arc uv in the ATSP
$c_{uv\kappa}^\delta$	Cost of traversing arc uv at instant κ in the ATSPTDC
$c_{uv\alpha}^\gamma$	Cost of traversing arc uv as the α^{th} arc in the TDTSP
c_{ijk}^ζ	Time to traverse arc ijk in the STDSP
C_{ij}	Cost of harvesting field i during period j in the DSS
d	Coefficient accounting for molasses value and processing difficulties in the RV payment system
d_ζ	End of a planning period in the STDSP
$d_{0,\ell_i(k)}^M$	Immediate event coefficient on yield
$d_{0,\ell_i(k)}^R$	Immediate event coefficient on RV
D_j^{max}	Maximum tonnage total for period j
D_j^{min}	Minimum tonnage total for period j
$\boldsymbol{e}$	Vector of residuals in regression
EJC	Number of ejection chains during tabu search
F_{ij}	Percent fibre content in field i during period j
$F_{i\ell_i(k)}^E$	Interaction coefficient between field facing direction and event
g	Number of attributes characterising a move in a tabu search
H_{ik}	Fraction of field i affected by event k
h_{ij}^ζ	Planning period during which vertex j is visited if visited after vertex i
I_{ijk}^ζ	An upper bound on the relevant time interval for arc ijk
ITR	Number of iterations of a tabu search
$L_{i\ell_i(k)}^E$	Interaction coefficient between field vertical location and event
λ_j^{poh}	Precipitation occurrences per hour
M	Problem-specific constant value

$\boldsymbol{M}$	Matrix containing M_{ij}-values
M_{ij}	Cane yield when harvesting field i during period j
M_1^{ζ}	First problem specific constant value in STDSP-formulation
M_2^{ζ}	Second problem specific constant value in STDSP-formulation
μ^{apa}	Dry-out rate
μ_j^{poh}	Dry-out rate during period j
N_{ij}	Percent non-sucrose content in field i during period j
$p_{uv\kappa}^{\delta}$	Waiting penalty for traversing arc uv early
p_{ij}^{ϵ}	Time to process job j on machine i in CJSP
p_j^{ζ}	Time to process job j in STDSP
P	Current recoverable value price per tonne
P_{ij}	Profit from harvesting field i during period j
P^E	Event penalty coefficient
P_i^W	Wet conditions risk penalty
ψ_j^{apa}	Amount of water per precipitation occurrence
Q	Vehicle capacity
q_u^{β}	Demand of customer u
$q_{ij\ell_i(k)}$	Time at the midpoint of period j since event $\ell_i(k)$ occurred on field i
$\boldsymbol{R}$	Matrix containing R_{ij}
r_{ij}	Resource required by agent i to perform job j
R_{ij}	Percent recoverable value in field i during period j
RND	Number of random restarts during a tabu search
S_{ij}	Sucrose content in percent of mass in field i during period j
r_{ij}^{ζ}	Restricted setup time between job i and job j
t_{uv}^{β}	Time required to traverse arc uv
$t_{uv\kappa}^{\delta}$	Time required to traverse arc uv starting at instant κ
t_p	Tabu tenure
t_{ij}^{ζ}	Departure time when exiting vertex j after visiting j
τ_u^{κ}	Time instants associated with arc u
τ^{ζ}	Starting time instant of a planning horizon
u_{ij}^{ζ}	Unrestricted setup time between job i and job j
Υ_{ij}	Performance parameter
V_{ij}	Value of field i during period j
$V_{i\ell_i(k)}^E$	Interaction coefficient between variety and event
$\bar{w}_j$	Average regional weekly RV during period j
$\boldsymbol{X}$	Matrix representing the design and model in linear regression
ξ	Fractional value of a decision variable during branching
$\bar{y}$	Average regional seasonal RV
$\boldsymbol{y}$	Vector of data in linear regression
$\hat{\boldsymbol{y}}$	Vector of estimated data in linear regression

CHAPTER 1

Introduction

Contents

1.1 Background

The problem of constructing a suitable schedule for the harvesting of sugarcane dates back to the period 6 000 BC to 500 BC, when the cultivation of this species of the grass family, *Poaceae* (earlier *Gramineae*), began in New Guinea and continued at a larger scale in India [62], having followed the human migration routes between Melanesia and South-east Asia [172]. The Persian Emperor Darius discovered "*the reed which gives honey without bees*" during his 510 BC intrusion into India Proper and brought it home to Persia, where honey was the only known sweetener at the time [189]. Alexander the Great discovered "*the sacred cane*" [210] during his two-year India campaign from 327 to 326 BC [210, 232], for the benefit of the Mediterranean nations [201].

It was, however, not until the Arab invasions of Persia in 633, 636 and 642 AD [229] that *Saccharum officinarum*, or "*noble cane*" (the sweetest species of the *Saccharum* genus), *S. spontaneum* (the species occurring in the wild), *S. barberi* (the most important species used in breeding) and *S. robustum* (the hard stalked species) [54] began to spread seriously, reaching Egypt in 710 AD [89], Spain in 755 AD, Sicily in 950 AD [15, 34] and was eventually brought by Columbus to Hispaniola on November 22, 1493 during his second journey to the Americas [230, 231]. In 1518, the Portuguese established the first sugarcane plantation in Brazil [218], today the largest sugar producer in the world, having a few decades earlier colonised some of the islands off the west coast of the African continent, namely Madeira and Sao Tome, successfully cultivating sugarcane on a large scale [62].

The southward spread of sugarcane cultivation and refining from Egypt—refining and re-crystallisation having been invented by the Egyptians [89]—to southern and eastern Africa occurred rather gradually during the period 1500–1850 AD. However, the Dutch introduced sugarcane to Mauritius prior to surrendering the island to the French in 1710 [151].

The first colonial sugar plantations in South Africa were established in Natal (currently KwaZulu-Natal) during the period 1846–1855 [10,197], the first plantation being *Compensation* in Umhlali established by a Mr Edmond Morewood who acquired his seed cane from Mauritius and Reunion [197]. Land had been opened to white settlers due to the annexation of Natal by order of the Cape Governor, Sir George Napier, on 31 May 1844 [185]. However, some claim that sugarcane was already grown by the Zulus at least as early as the 17th century [10].

From the 1850s onward, sugarcane growers established themselves throughout Natal, some immigrating from Europe or Mauritius, lured by word of profitability stemming from initial success [197], while many workers were "recruited" as indentured labourers from India during the 1860s [196]. The general trend during the 20th century was for sugarcane farms to expand, the initially small, scattered mills to be consolidated and for large sugar companies to emerge. Some farms remain small to this day, other farms have merged into large, family-owned farms while still others have become incorporated into commercial estates. This relative diversity today makes up the fourteen established mill supply areas from Mpumalanga Province in the north to southern KwaZulu-Natal, contained within the areas indicated in Figure 1.1. The approximately 47 000 sugarcane farms in South Africa [194] currently produce more than 20 million tonnes of sugarcane per annum, resulting in approximately 2.3 million tonnes of sugar each year [108].

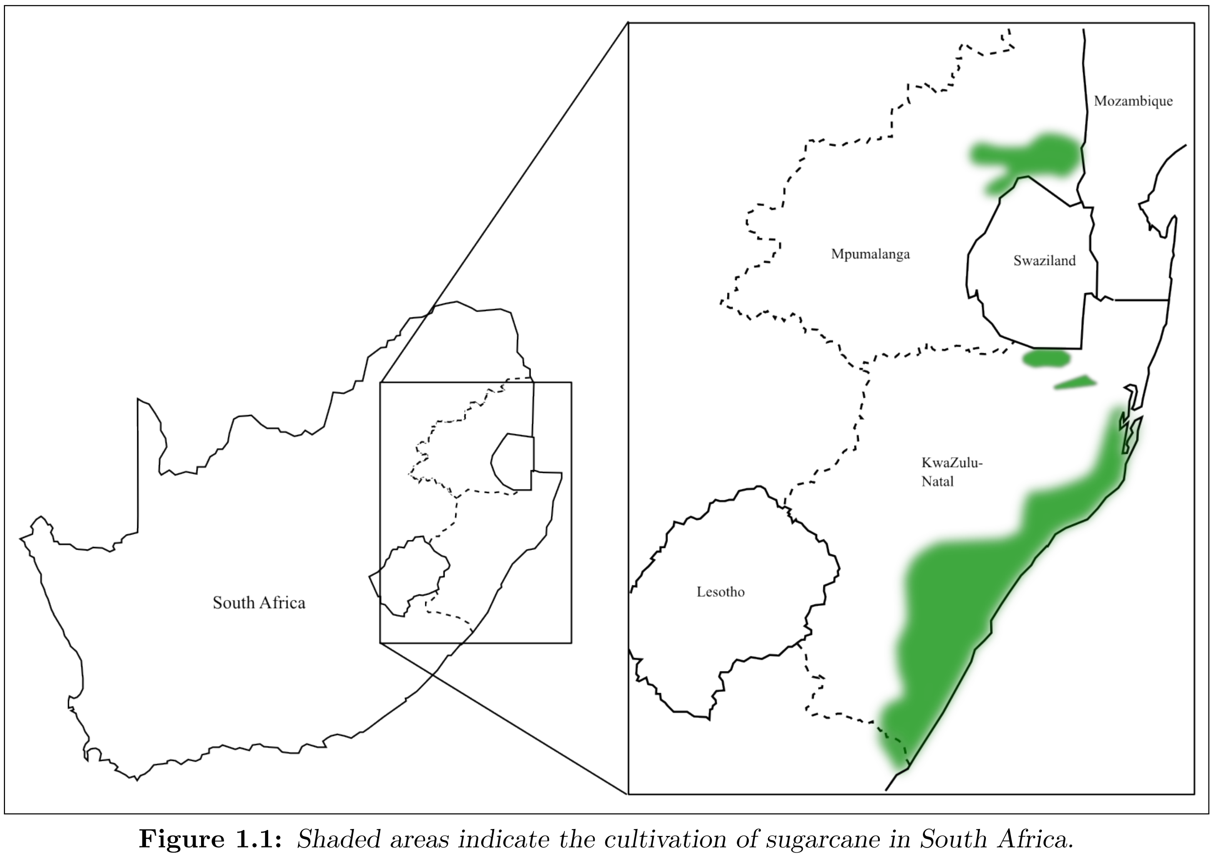


Figure 1.1: *Shaded areas indicate the cultivation of sugarcane in South Africa.*

1.2 Informal problem description

Increasing global competition in the international marketplace, mainly spearheaded by the Brazilian and Indian sugar industries, results in sugar prices remaining relatively low worldwide [55]. However, increasing fuel prices, perhaps currently driven by the rate of depletion of fossil fuels and the emerging industrial giants, China and India [228], are raising operational costs of farming. A mill supply area may comprise several thousand subsistence farmers, several hundred commercial growers and a few very large estates [194]. The commercial growers and large estates are, from a purely economical perspective, more important. For example, 83 % of the sugarcane milled in South Africa by Tongaat Hulett Sugar, one of the large sugar companies, is grown on company land or large commercial farms, while only 17 % is grown on small and medium-scale farms [215]. Increasing pressure from the global market on the South African sugar supply chain [135] combined with the fact that sugarcane is an age-deteriorating product which loses value while in the transportation stage [191], leads to the requirement that the logistics of the sugar supply chain be streamlined in order to ensure the survival of the industry. Indeed, there is currently a concerted effort in the industry to collaborate across business entity boundaries and with government [193]. For example, there is collaboration among growers with respect to co-owning logistical equipment [25] and collaboration between growers and millers concerning the implementation of computerised logistical scheduling systems [69]. In terms of computerised decision support for sugarcane estates, SQR Software's CANEPRO system [199] supports the management of 130 000 hectares of land for several estates with respect to operational planning, manual agricultural activity scheduling, field data-basing and costing.

A farm of commercial proportions is ideally designed and managed to maximise land use, providing fields that are fit for available cultivation and harvesting equipment as well as accessible by the relevant transportation vehicles, while minimising negative environmental impact such as erosion, wetland encroachment and flora and fauna habitat destruction [181,182]. One visible effect of the layout design is that farms are typically divided into fields, usually separated by relatively wide firebreaks, neighbouring fields often differing quite substantially in appearance from one another. These differences in appearance are not circumstantial; it is due to careful planning of cultivation stages, crop age[1] and sugarcane variety interspersion, so as to minimise the risks of runaway fires, insect infestation proliferation, the spread of fungal and other diseases as well as hampering hillside water run-off velocities during periods of heavy rainfall [182].

While layout design is an ongoing issue at the *strategic* farming level, one may argue that *harvesting tactics* are in themselves crucial in maintaining a resilient and sustainable commercial sugarcane farm—both financially and environmentally. Even with a carefully planned layout, some fields may be connected by roads that are too steep to travel on or may themselves be too steep to allow harvesting during the rainy season. Figure 1.2 shows sugarcane harvesting in progress on a hillside. Other fields may be situated in an area that is exceptionally dry, experiences frost regularly or dries off slowly following rain. While considering such physical properties of the farm, it is equally important to be aware of any occasional minor disruption or major disaster that may strike the sugarcane. A frosted field, or one that has been overrun by accidental fire, may become a financial disaster if not tended to immediately, while a field that has had its sugarcane lodged (blown down) may only constitute a developing problem [182].

In South Africa, sugarcane is grown in cycles of twelve to twenty-four months. Different sugarcane varieties display different seasonal sucrose content trends, which generally peak during the middle of winter [182]. Ideally, this is the time around which the entire farming strategy, layout design, sugarcane variety interspersion planning and harvesting tactics should be oriented. The tactical harvesting problem is, however, a more challenging reality since cane must be harvested at a consistent rate throughout the entire nine-month season, according to the rules and regulations governing the South African sugar industry. This is to spread out the workload, requiring

[1]These differences in age is called the *age-mosaic*.

Figure 1.2: *Hillside harvesting should preferably occur during dry conditions, since harvesting machinery tends to slide on wet, sloping earth. The topography of a farm is one of the factors to consider in tactical sugarcane harvest scheduling.*

less milling capacity and thus save money.

A commercial sugarcane farm may comprise more than 120 fields, and most of the crops on these fields as well as the field properties themselves, are different from one another. These differences may be small when viewing them at an instant, but their effects typically become more distinguishable over time. When considering the number of different answers to the question of how to schedule the harvest of 120 different fields, the harvest scheduling problem seems a little daunting. Determining the harvest schedule is, however, at the centre of remediating the negative of the effects of the various disruptions and major disasters mentioned earlier and profiting from the positive of these effects; hence the challenge and value embedded in the *tactical sugarcane harvest scheduling problem* treated in this dissertation.

This dissertation contains an approach to modelling and solving the problem of providing computerised decision support to people charged with the task of scheduling the harvest of sugarcane. The approach involves models of sugarcane growth, sugarcane deterioration when struck by disease, frost, insect infestations, fire and other adverse events. These models are then used to estimate the net profit from harvesting each field at each point in time into the future of a current season. Once each field has a net profit associated with its harvesting, an optimisation model is used to construct a schedule of maximum estimated profit, by ordering the fields in such a way as to maximise their combined net profit. This schedule is the proposed solution to the decision support problem described in this dissertation, and the focus is mainly on medium-scale commercial growers and groups of such growers working co-operatively.

1.3 Dissertation aim and objectives

This dissertation is aimed at advancing the current state of research on *decision support systems* (DSSs) for sugarcane harvest scheduling in South Africa. Towards realising this aim, nine detailed objectives are pursued throughout.

Objective I: To *perform* a literature survey of operations research models previously formulated in the context of optimising the sugar supply chain, the sugar supply chain's individual parts from growing to the mill yard, and other supply chains that relate to the problem at hand.

Objective II: To *perform* a literature survey of related combinatorial optimisation problems and conventional solution methodologies in order to establish a foundation of scientifically sound and practically suitable models and associated solution methods on which to base the core of the DSS put forward in this dissertation.

Objective III: To *perform* a literature survey of the methods previously and currently used to model sugarcane mass, value per unit of mass and harvesting costs in order to establish a foundation from which to *develop* a suitable methodology for predicting

a) sugarcane yield based on readily available data,

b) sugarcane relative recoverable value percentage based on readily available data,

c) sugarcane harvesting costs based on readily available data.

Objective IV: To *perform* a literature survey and to *interact* with suitable role-players in the sugar industry in order to gain an understanding of the various factors that affect the growth, quality and survival of sugarcane crops, in order to establish a foundation on which to *develop* means of adjusting predictions of sugarcane yield, relative recoverable value percentage and harvesting costs, when such factors are in effect. Examples of these factors are foreseen to include, but are not limited to, accidental fire, frost of varying degrees, diseases, insect infestations and drought.

Objective V: To *conduct* an empirical combined evaluation and development experiment at the level of potential DSS users in order to *determine* what capabilities a tactical harvest scheduling DSS should possess and how those capabilities may be supported mathematically, in order for such a DSS to be useful to the South African sugar industry.

Objective VI: To put forward a viable suggestion with respect to sugarcane crop harvesting decision support, based on the results of pursuing Objectives I–V, by

a) *designing* a generic DSS architecture so that it may accommodate relevant building blocks without loss of capability, showing its data requirements, how the data are transformed and flow within the DSS, as well as what processes utilise the data and how these processes are sequenced,

b) *populate* the DSS architecture with various building blocks, such as interfaces, databases, mathematical models, solution methodologies for these models and evaluation tools.

Objective VII: To render useful the DSS of Objective VI, by

a) *implementing* the DSS on a personal computer so as to achieve a stand-alone program which does not require specialist software to be installed for its execution and subsequent use,

b) *testing and debugging* the DSS implementation by means of a sequence of experiments interspersed with implementation corrections designed to render the DSS capable of performing consistently and predictably in practice.

Objective VIII: To *validate* the effectiveness of the DSS implementation of Objective VII, by

a) *shadow scheduling* at least one actual harvesting operation for an entire season, while implementing newly found necessary or improving changes or additions to the various building blocks of the DSS,

b) *drawing* opinions from the industry representatives involved in a),

c) *comparing* actual harvesting schedules with the DSS-generated schedules as a means of benchmarking the DSS with respect to the decisions of industry professionals.

Objective IX: To *outline* ideas for future work and to provide possible directions in which to proceed in order to further advance the research on operations research-based DSSs for tactical sugarcane harvest scheduling in South Africa and other nations.

1.4 Dissertation scope

The methodology within the DSS for predicting *cane yield*[2] for an undamaged, disease-free cane crop not treated with *chemical ripener*[3], not in a state of water stress, shall provide for the employment of empirical models derived from data readily available to the *user*[4], (such as field records from a representative geographical area under representative *agroclimatic*[5] conditions). The DSS shall provide for the user to be able to manually adjust the predicted cane yield values field-by-field for perceived errors due to detailed conditions which are in this dissertation considered unpredictable. For example, such conditions may be unexplained low cane yield in a field which is apparently healthy, low cane yield in a field due to *crop class*[6] (when data are not sufficient for incorporating the factor crop class into the prediction models) or generally different than expected cane yield in some field due to other factors that were not modelled explicitly when applying the methodology for predicting cane yield on the available data.

The methodology within the DSS for predicting *cane recoverable value percentage*[7] (RV %) shall be subject to the same limitations and provision requirements as the methodology within the DSS for predicting cane yield, insofar as these limitations and provisions bear meaningful implications, except for the provision that the user may manually adjust cane yield estimates, but not cane recoverable value percentage estimates.

The DSS shall accommodate the incorporation, by the user of the DSS, of alternative models for predicting the effects of *extraneous events*[8] that may impact substantially on the value or future value of cane crops[9], and in particular their effects on cane yield and cane recoverable value percentage as functions of time (since the occurrence of the event).

The DSS shall take into consideration to a reasonable level the differences in harvesting costs which arise due to varying field properties and weather. It shall be possible for the user to

[2]The term *cane* is taken throughout this dissertation to mean sugarcane. The term *cane yield* is taken throughout this dissertation to mean the mass of cane, including the sucrose and other components, and is thus a production quantity. The term *cane yield per hectare* is taken to mean the mass of cane per hectare and is thus a measure of productivity.

[3]A *chemical ripener* is a chemical that, when applied to unstressed sugarcane, may increase the amount of sugar in the cane stalks as well as the purity of the juices within the cane stalks [38].

[4]The term *user* is taken throughout this dissertation to mean a person who knows how to use the DSS put forward in this dissertation.

[5]The term *agroclimatic* is used throughout this dissertation to mean temperature, incident solar radiation, rainfall and wind conditions; climatic factors that are known to influence agriculture [18].

[6]The term *crop class* refers to the number of times that a cane field has been harvested since it was planted. A crop class of 0 is called a *plant crop* and a crop class of 1 means that the crop has re-germinated from the root system of a plant crop after initially being harvested, such a crop is also called a *ratoon.*

[7]The term *cane recoverable value percentage* means the percentage of the cane yield of a crop for which the seller of the crop receives payment, after the affixation of a market and regulation-dependent *recoverable value price* [195].

[8]The term *extraneous event* is taken throughout this dissertation to mean an event, which may occur on or inside the cane roots, stalks or leaves as well as in a crop or field as a whole, and which is known to alter the normal growth or normal quality of the crop positively or negatively.

[9]The term *crop* is taken throughout this dissertation to mean the sugarcane present in a single field. The term *field* is therefore equivalent to the term *crop* when referring to the sugarcane, but the term field is more appropriate when referring to the field as part of the layout of the farm.

adjust these considerations to fit his/her situation.

The DSS shall be designed and validated for South African conditions, for geographical and agro-climatic regions where sugarcane crops are harvested approximately every twenty-two months and do not require extensive irrigation.

Furthermore, the DSS is to be designed for medium-scale growers and co-operative joint ventures between such growers, but is hoped to apply to other types of sugarcane harvesting contexts as well.

1.5 Dissertation structure

This dissertation comprises three parts: a part on previous work by researchers related to sugarcane production leading up to a formal problem description, a part dedicated to describing the solution to the problem and finally a part devoted to summarising the dissertation and presenting a few proposals for future research.

Part I consists of Chapters 2, 3 and 4. Chapter 2 contains a brief review of the literature on previous work employing operations research-based methods on modelling sugarcane flow, a very brief review of current research in the field of supply chain management and a brief review of value chain research on agricultural value chains. Moreover, Chapter 2 contains an effort to organise previous results on how sugarcane deteriorates when having been subjected to various extraneous events, and is in particular devoted to a thorough study of authoritative information about such events that is readily available to any sugarcane grower in South Africa. Chapter 3 contains a description of the sugar supply chain from the planting of sugarcane to the delivery of sugarcane at the mill. The chapter is dedicated to presenting a brief overview of some main activities that impact the grounds on which the harvest scheduling decisions are normally taken. The final chapter of the first part, Chapter 4, contains a more formal description of the problem at hand. The problem is conceptually placed within several contexts and formulated within each one.

Part II of this dissertation consists of Chapters 5, 6, 7, 8 and 9 which together describe a solution to the problem at hand. Chapter 5 contains a brief review of the literature on combinatorial optimisation problems and combinatorial optimisation solution methodologies relevant to the problem at hand. The problems and solution methodologies considered are here described only in order to bring the problem into a combinatorial optimisation context. Chapter 5 is thus together with Chapter 4 the formative chapters in defining the problem. Chapter 6 is committed to the description of two different formulations employed to model the problem and some solution methods employed in order to solve it. The two formulations in Chapter 6 require parameters and data, and the requirements on these are described in Chapter 7 which also contains descriptions of the methodologies employed to generate these parameters from available data as well as descriptions of how the parameters are turned into the correct form implied by the optimisation model formulation. Chapter 7 also contains conclusions drawn from Chapter 2 with respect to growth and deterioration related parameters. The models and their parameters are placed in a common architecture which is described in Chapter 8. In Chapter 8, there is a description of this DSS architecture and its building blocks and how the DSS may be implemented on a personal computer. A description of activities and skills required if installing and deploying this computer implementation in a real problem environment is presented and the various implemented building blocks are shown by means of screen-shots. There is a section dedicated to verifying that the computer implementation functions as intended and a series of experiments are performed in order to determine relevant settings for solution method parameters. In Chapter 9, the validation and development experiments are described.

Part III contains the conclusion of this dissertation in Chapter 10 and an outline of proposed future work in Chapter 11. Chapter 10 consists of a comprehensive summary of the dissertation, where each chapter is mentioned and reference is made to the dissertation objective fulfilled by the work contained within the chapter. The main contributions of this dissertation are then outlined and subsequently appraised. Finally, Chapter 11 is dedicated to several proposals for future research and each one is described briefly.

Part I

Sugarcane production

CHAPTER 2

Literature review

Contents

This chapter contains an overview of the relevant literature with respect to sugarcane production, aimed at providing a basis for developing the necessary modelling methodology within the DSS framework presented later in this dissertation.

2.1 Introduction

Sugarcane production has been modelled using many different approaches, such as mathematical programming for scheduling, planning or optimising sugarcane supply, discrete event simulation and queueing theory to improve mill-yard operations and many other modelling techniques for the vast array of problems arising in this context. The following phenomena receive special attention during this literature review: operations research (OR) approaches to sugarcane modelling, live sugarcane crop quality and yield models, cut cane stalk effects, burnt cane stalk effects, frosted cane stalk effects, effects of lodging and drought effects. Research areas that relate to the framework of the problem include supply chain management philosophies and value chain research in agriculture. Even if these research areas are not related to sugarcane harvest scheduling it may be beneficial to review some current advancements in the general area of supply chain improvement work.

2.2 OR approaches to modelling sugarcane flow

Grunow *et al.* [81] considered parts of the Venezuelan supply network, where most farms (haciendas) are owned by the same business entity that owns the mill. According to Grunow *et al.*, there is no paper in literature that deals with decisions across levels from strategic through tactical to operational. The main objective of their proposed mixed integer linear programming (MILP) model was to secure a minimum supply of cane to the mill. Their step-by-step, hierarchical modelling and solution procedure starts at the strategic level by assigning a cultivation date to each hacienda, after which the solution for that model is implemented. When it is time to start harvesting, a tactical level model is solved in order to determine optimal haciendas to start harvesting for every day for the following two weeks. This harvesting schedule is used at a third modelling level to generate crew and equipment dispatch schedules. One assumption in their model is that once the harvest of a particular hacienda has begun, it only stops when completed. This constitutes an important difference between the South American and South African harvesting circumstances, since in South Africa farms are not by rule harvested in a single effort, but rather on a field-by-field basis with the harvesting effort tempered to adhere to a *daily rateable delivery*[1] (DRD). In addition to securing supply to the mill during normal conditions, Grunow *et al.* claim to address the occurrences of unforeseen events such as criminal fires by accommodating data changes at the tactical level (harvest day).

A modelling approach by Higgins *et al.* [101] and Muchow *et al.* [158] incorporated six years of block productivity data in an attempt to optimise the harvest schedule of an entire mill region in Australia. The model parameters were supported by a number of one-way *analyses of variance* (ANOVAs) which were used to find significant differences in sugar yield due to geographical, cane variety, crop class and other circumstances. The authors of [101] concluded that there is opportunity to increase the profitability of the investigated mill region as a whole.

Higgins [97] adopted a MILP formulation towards modelling the transportation problem of sugarcane in Australia. He emphasised the importance of using a participatory research approach which he attributes to Martin *et al.* [148]. The model was used to some extent by transport planners in the Maryborough mill region.

A review by Weintraub and Murray [227] shows how the forest harvest scheduling area addresses many of the issues found in sugarcane harvest scheduling. The mathematical formulations presented by Weintraub and Murray include retaining a "mosaic" of forest patches analogous to retaining a healthy age or variety difference between adjacent sugarcane fields. The best solution approach towards solving real problems, according to Weintraub and Murray, are metaheuristics such as genetic algorithms, tabu search and simulated annealing. A model by Karlsson *et al.* [117], similar to the models reviewed by Weintraub and Murray, was designed to schedule harvesting crews while obeying a number of demand, capacity and transportation (road)

[1]The term *daily rateable delivery* is taken throughout this dissertation to retain the same meaning as the term *rateable delivery* defined in the *sugar industry agreement* (SIA) [49]. The SIA term means that the total cane delivered by any grower shall be delivered to the mill in a rateable manner throughout the entire season.

constraints. Karlsson *et al.* compared a number of solution approaches and also preferred a metaheuristic approach. Rönnqvist [177] gives an overview of optimisation problems found in the forest industry.

Guilleman *et al.* [83] investigated several mill-scale harvesting schemes using CIRAD's[2] simulation tool named MAGI[3] [134] to determine whether harvesting sugarcane without the requirement of adhering to the DRD could increase revenue for all parties at the Sezela mill in South Africa during the 2000 season. Results indicated gains in RV % valued between approximately 4–15 million Rand, depending on scenario. The study was followed up by an investigation by Le Gal *et al.* [135] addressing essentially the same question, assuming that RV-prediction functions change significantly from year to year. Their conclusion was that the Sezela mill region could gain up to 7.44 million Rand, given an average sugarcane crush rate of 2.23 million tonnes per year. The authors in [135, p. 66] suggest several general views on modelling sugarcane supply, and claim that

> "These optimisation models are not directly applicable in the South African case. Firstly, they need a detailed data set including the field level. Secondly, they assume either that a central controller implements the calculated solution by deciding precisely which fields have to be harvested within the season and by allocating both harvest and transport capacities between farms, or that all the stake-holders agree with the optimal solution and are ready and able to implement it at their own level. Thirdly, these models are based on complex algorithms which make them difficult to explain in a decision support perspective. Fourthly, they give little room for discussion or compromise as they provide 'one best way' solutions."

In 2004, Higgins and Postma [102, p. 237] pointed out the importance of allowing each grower fair harvesting time windows throughout the season, which "*. . . prevents any grower from being unfairly exposed to wet weather risks towards the start or the end of the harvest season, as well as other seasonal effects.*" The results of Higgins and Postma's work on the optimisation of siding rosters[4] which entailed a solution to the *staff* roster problem, was implemented by the participating mill and cost savings were estimated to approximately 150 000 Australian dollars/year. Higgins and Postma [102, p. 248] claim—in obvious contrast to the above quoted Le Gal *et al.* [135, p. 66] claim, that

> "Given the complexities of producing a staff roster acceptable for use in the Australian sugar industry, we would argue that traditional scientific research and development principles would not have achieved implementation. The siding rostering model of this paper is adaptable [to] all other sugar mill[s] in Australia and any sugar producing country that uses mechanical harvesters, such as Brazil, South Africa and the United States."

In a 2002 paper, Higgins [96] placed focus on the participative nature required to gain success in developing and implementing operations research-based optimisation models or tools for the Australian sugar industry. This time Higgins developed an optimisation model for mechanical harvester scheduling, which eventually led to the highly successful work on the optimisation of siding rosters mentioned above.

[2]Centre de Cooperation Internationale en Recherche Agronomique puor le Développement.

[3]The meaning or origin of the acronym is not explained in [134].

[4]Siding rosters are the schedules used by Mackay Sugar to distribute time-slots for cane deliveries to railway sidings throughout the mill-region.

Iannoni and Morabito [106] formulated a discrete event simulation model of the mill yard of a large sugarcane mill in Brazil, assuming that the arrival rate at the mill is a result of the number of long-range transport vehicles in the system combined with the transportation vehicles' cycle time. The cycle time was defined as the sum of waiting and unloading time in the mill yard and the time to travel to the field, load and return to the mill yard. The authors ran three scenarios which showed promise in that it was, according to the model, possible to decrease the amount of sugarcane in queue and to increase the utilisation of the mill. The scenario that showed promise consisted of changing dispatching and queueing rules for vehicles of a certain type, effectively increasing the amount of unloaded sugarcane. The model represents the sugarcane supply system, which is, according to Iannoni and Morabito, a closed system since long-range transport vehicles loop. A problem with this approach, should it be applied to the South African case, according to the author of this dissertation, is that a closed system assumes that the amount of sugarcane available for pick-up per time unit is governed by the cycle time of the long-range transport vehicles. A more accurate way to model the available cane for the South African case is to consider the mill crush rate, the DRDs of all growers and the cutting capacity of all growers. However, Iannoni and Morabito provide validation of their model towards answering the questions they put forth.

Higgins and Davies [100] built a stochastic simulation model for the purpose of long-term transport system capacity planning and considered several scenarios developed together with representatives from growers, harvesters and the miller in the Mourilyan mill region in Australia. One of the scenarios was deemed promising and was later implemented, the results of which were evaluated by the group of representatives to be of benefit to the miller, while growers and harvesters were not negative. In the same paper, Higgins and Davies developed a second simulation model to optimise the starting time for each harvester in the promising simulation scenario. The new harvester schedule indicated a reduction of the 1.5 hour average waiting time at the railway sidings for harvesters by 70 %. The new harvesting starting times schedule was adopted by the group of representatives in 2004, not without harvesters being unhappy with the optimised schedule, raising some objections.

Le Gal *et al.* [133] presented a method for conceptually integrating three levels of modelling of the sugarcane supply chain, namely the operational, tactical and strategic levels. The framework that they proposed is an idea of how to combine the three software packages CIRAD's MAGI [134], Rockwell Automation's ARENA [7] and Weintraub and Epstein's decision support system for forestry transportation scheduling ASICAM [226] and constitutes, according to the authors, a step towards an operational support method. Grunow *et al.* [81] as well as Le Gal *et al.* point out that little or no research is aimed at addressing decision support on all levels across the supply chain in a single approach. The author of this dissertation believes that there are no such actual decisions to be supported, mainly since the ownership structure of the sugarcane supply chain is too diverse, at least in South Africa. Before models can *really* be applied to all three levels of the same supply chain, the various owners must somehow embark on joint ventures under which such decisions are actually likely to occur. The highly successful project during which the decision support system ASICAM was developed was, incidentally, a project where such decision existed [226].

In Thailand, a recent effort by Piewthongngam *et al.* [170] points out that communication between growers and millers in the northeast of Thailand is poor. Here, the amount of cane delivered each day is not determined by contract as in South Africa, which leads to problems such as under-utilisation of the mill and sugar losses. Piewthongngam *et al.* developed a framework for cultivation planning on the field level, in order to provide possible alleviation of the problem, using crop growth simulation models for a yield estimation module and mathematical programming for a planning module. The mathematical formulation approach is to maximise sugar production by mass, subject to a set of constraints ensuring that the available area of land is not exceeded and a set of constraints ensuring that the milling capacity is exactly met. The Piewthongngam framework considers an entire mill region at once, attempts to make field level decisions one season into the future and has yet to be validated through actual trials.

2.3 Supply chain improvement methodologies and philosophies

According to Stock [208, p. 147], there are three main areas of *supply chain management* (SCM) research in the literature:

> "...(1) development of methods and techniques to study SCM and its components/processes; (2) developing solutions or answers to specific supply chain-related problems or challenges; and/or (3) measuring the results or outcomes of supply chain strategies and tactics."

He furthermore states (in [208, p. 156]) that more theoretical work is needed for SCM research to develop, but mentions that there is some consensus around the notion of the supply chain modelled by

> "...four levels of analysis: process types (plan, source, make, deliver, return); process categories (types of processes, such as planning, execution, enable); decompose processes (definitions, source of inputs and outputs, performance metrics, best practices); and decompose process elements (*i.e.* implementation)."

Kleijnen and Smits [121] describe the quantification of supply chains and discuss four different simulation-style analysis tools. They argue that an organisation should measure its own performance, not the performance of the entire supply chain, but should inform itself of and be part of the general striving of the rest of the value net. Kleijnen *et al.* in essence suggest a four-step approach to supply chain research: finding a subject *supply chain* (SC), deciding upon and defining metrics, building a simulation model, and finally performing a sensitivity, optimisation and robustness analysis using the simulation model. Kleijnen's *et al.* proposed methodology could be compared to what is used in Six Sigma (invented at Motorola in 1996) called the *design, measure, analyse, improve, control-cycle* (DMAIC) [114]. The DMAIC is a five-step approach to problem solving using a toolbox that comprises essentially any mathematical, statistical, visual or other quantitative tool that is approved by management. The user first *defines* the problem, then determines a way to *measure* it, then *analyses* it to find the root cause(s), then *improves* (*i.e.* develops and implements a solution) and finally *controls* the implemented solution until the situation has stabilised and become part of the daily operational procedures or protocols.

Within the school of thought of SCM, however, a vast amount of research exists on the subject of supply chain *performance measurement frameworks* (PMF). An example is Beamon [11], who in 1999 developed a framework by which to choose measures for improving manufacturing supply chains. Beamon defined a number of concrete measures, which were divided into three main groups, namely, resources, output and flexibility. By categorising any set of performance measures into these three (employing the framework), Beamon claims that organisations may more completely characterise their supply chain. Gunasekaran *et al.* [84, p. 334] stated:

> “Furthermore, computer generated production schedules, increasing importance of controlling inventory, government regulations and actions such as the creation of a single European market, and the guidelines of GATT and WTO have provided the stimulus for development of and existing trends in SCM.”

Gunasekaran *et al.* thus support computer generated production schedules as part of the supply chain improvement mission. Their framework consists of crossing four—as opposed to Beamon’s three and Stock’s five—categories, namely plan, source, make/assemble and deliver with either the strategic, tactical or operational level. A measure may, for example, be categorised as source/strategic if it concerns the *lead time performance* (strategic measure) of a *supplier* (source level) compared to the industry norm [84, p. 336].

It is agreed within the supply chain management community that today companies must cooperate, which is expressed as follows by van Hoek *et al.* [222, p. 126]:

> “One [lesson] is that companies have to align with suppliers, suppliers’ suppliers, customers and customers’ customers to streamline operations. As a result, increasingly supply chains are the dominant vehicle for competition and not individual companies. Another lesson is that within the supply chain, companies should work together to achieve a level of agility beyond the reach of the individual company.”

The use of PMF is generally regarded as fruitful but not simple to implement. Beamon [11] and Gunasekaran *et al.* [84] share the idea put forth by others, such as Gunasekaran and Tirtiroglu [85], Gilmour [70], and van Hoek [220], that performance measurement can drive and guide improvements. In [221], van Hoek used performance measurement to strengthen the hypothesis that companies will ally more horizontally in the future. According to Hervani *et al.* [93], performance measurement may be used for environmental purposes. Chan and Qi [30, 31] contributed with a measurement algoritm as a framework and Aramyan *et al.* [6] used PMF in the agricultural-food supply chain (in their case tomatoes). Bichou and Gray [19] expressed difficulty in applying supply chain PMF to the port industry, and notably point to the same research methodology as did Higgins [96], namely *participatory research*. Lai *et al.* [128] developed a framework for use in the transportation industry and Swinehart and Smith [211] used *total quality management* (TQM) to design a continuous improvement programme for health care supply chains, including performance measures. Swinehart and Smith may not be the first to connect PMF and TQM, but did so very clearly, showing that different schools of thought in terms of management should not be regarded as alternatives, but rather complementary.

2.4 Value chain research in agricultural supply chains

In 2005, the Australian Sugar Research and Development Corporation completed an investigation into the opportunities for future research on the Australian sugar value chain [203]. The inquiry was based on a once-off forum comprising 80 participants representing all areas of the industry and a report written by Higgins *et al.* [99] on a study of various R&D projects which had taken place in the Maryborough region during the period 1995–2005. In these value chain R&D projects, the results were in part due to the participatory research approach, implying that the intangible output of the work itself is as important as the planned goals and outcomes. Comments from participants were positively directed towards harvesting and transport scheduling, *e.g. “Harvest and transport scheduling work has had the biggest advantages; it has highlighted the need to be flexible about how to organise harvesting in the region and has provided figures to* *work out how to organise the harvesting system more efficiently"* [203, p. 14] and so contradicts statements by Le Gal *et al.* [135, p. 66] claiming that: *"... these optimisation models are based on complex algorithms which make them difficult to explain in a decision support perspective."* Le Gal is correct in that solving scheduling problems involves complex algorithms, but that does not mean that users cannot understand the solutions. One example occurs in the trucking industry, where large as well as small trucking companies employ difficult-to-understand algorithms to produce easy-to-understand solutions to their various scheduling and routing problems.

Findings during R&D in the Mackay region furthermore suggests that *"To date* [cane supply scheduling] *has been adopted mainly at the block or farm level so impact has been limited. However, knowledgeable industry leaders believe that the time will come when the industry will adopt it at the regional level and reap substantial benefits"* [203, p. 17]. A successful, large scale attempt to implement and integrate several supply chain models (and single unit models) by Higgins *et al.* [98] revealed how important it is for the researcher to work closely with the subject in order to gain acceptance for methods used and in order for the subject to understand and use the solutions. It seems, according to the author of this dissertation, that the manner in which a particular methodology or tool is implemented speaks more about how well it will be understood, than the inherent properties of the tool itself. Obviously, the solution that the tool provides must in both cases be beneficial, in order to be accepted. For further support of the participatory approach, see Higgins *et al.* [103].

Other research that has examined the whole of the value chain (which is the same as the supply chain, but often points solely at the valuable materials or information that flows through the chain) includes a paper on suitable modelling approaches for inter-company cooperation by Gaucher *et al.* [66], a web-based cooperation platform for the Australian Mackay mill region developed by Fleming [58], an evaluation by Thorburn *et al.* [213] of a whole-region value chain for the purpose of cogeneration and an inquiry into the potential for dimethyl ether production throughout the biomass flow of the sugarcane value chain by Chohfi [36].

2.5 Sugarcane growth models

In his review of sugarcane simulation models in 2000, O'Leary [163] argued that there are two simulation-based growth models in use in the world today, namely APSIM Software Engineering's APSIM-Sugarcane by Keating *et al.* [118] and the South African Sugarcane Research Institute's CANEGRO by Inman-Bamber [109] and Inman-Bamber *et al.* [110]. O'Leary [163, p. 98] furthermore suggested that regression-based models are highly prevalent in "*site-specific studies.*" APSIM-Sugarcane is, according to O'Leary, unable to accurately simulate sucrose accumulation under significant water stress, but is otherwise able to predict yields with a root mean squared error (RMSE) of 1.97 t.ha^{-1}. CANEGRO performs at an RMSE of 0.95 t.ha^{-1} while a less known model by the name of QCANE outperformed both of these well-known models with an RMSE of 0.85 t.ha^{-1} [163]. QCANE was developed by Liu and Bull in Australia [143] and is being further developed (see *e.g.* [144]).

There are two major models in use in South Africa for predicting the yield and sucrose content of sugarcane based on environmental and weather data. These models are CANEGRO and the South African Sugarcane Research Institute's CANESIM [16, 17]. According to Bezuidenhout *et al.* [18], CANEGRO is not suitable for estimating RV since it does not distinguish between the cane constituents non-sucrose and fibre. CANESIM consistently over-estimates cane yield per hectare and also does not directly report recoverable value percentage as an output. The claim with respect to over-estimation follows from the validation run in [16, 17] on fifteen mills and data from 1980 to 2002. The level of over estimation is estimated at an average of 50 % [16,17]. Moreover, [16, 17] claims that the model will outperform the Mill Group Boards[5] early in the season, but that the opposite is true later in the season.

Instead of relying on relationships between rainfall, soil, yield and RV %, albeit well-known, yield and RV % may be modelled using multiple regression. Using a database of yield data from multiple states during the period 1963 to 2002, maintained by the American Sugar League, Greenland [79] performed a statistical analysis in 2005. The resulting four variable model reportedly modelled the yield with an RMSE of 5.1 t.ha^{-1}. The model variables in Greenland's model were selected through ranking the correlation coefficients between 74 climatic variables and the response (the dependent variable), yield, selecting the variables with the largest coefficients (while excluding some of the intercorrelated variables). The yield that Greenland attempted to model is the grand average of a season for an entire state and the RMSE value essentially means that the model could predict the average yield of the state to within $\pm\ 5.1 \times 1.96 = 10.0$ t.ha^{-1}, assuming that the residual errors in his model are normally and independently distributed (NID). Jiao *et al.* [116] also used historical data to model cane productivity; in their case the response variable was the Australian basis for cane payments, *commercial cane sugar* (CCS). Incidentally, the authors in [116] used regression models to optimise harvesting of fields tactically across multiple farms. They optimised harvest schedules across the season for multiple farms within a mill region in Australia. The approach is focused on maximising sugar output from harvesting groups. Jiao *et al.* used a linear programming model to find the optimal proportion of each farm to harvest during each harvesting round and also to determine during which week each harvesting round should take place for each farm. The prediction functions were found by fitting a one-factor second-order polynomial using linear regression analysis. The mill area seasonal average CCS prediction problem using historical productivity data in Australia was also approached by Lawes and Lawn [132]. Lawes and Lawn mention that the variable *farm* may constitute a *surrogate variable* for several hard-to-capture variables that are more traditionally used in explaining yield and CCS. The same type of data, called *block productivity data*[6], were used in 1998 by Higgins *et al.* [101] and later in 1999 by Higgins [94] to build regression models feeding into mathematical programming models which optimise harvest scheduling across entire mill regions. The regressors harvest time, harvest age, crop class and farm paddock (field) were connected to the responses yield and CCS.

Park *et al.* [165, p. 318] found that the *radiation use efficiency* (RUE) of sugarcane sometimes decreases dramatically and that the phenomenon requires further study; they specifically suggested multivariate analysis to assess "*a broader range of variables than is currently available is necessary if we are to ensure that harvesting schedules based on maximising whole of industry profitability, realise their full potential.*" Park *et al.* mentioned lodging[7] as a factor.

Singels *et al.* [187] performed three trials designed to study the effects of crop class, start date (previous harvest date) and variety on the sugarcane plant's ability to absorb light. Singels *et al.* argued that by understanding light absorption, cane yield improvements will ensue. Singels's *et al.* three trials showed that canopy[8] development during the initial approximately 100–300

[5]Mill Group Boards are boards of representatives from growers and millers who are charged with the task of "... [providing] *services aimed at facilitating the reception and testing of cane delivered to the mills concerned...*" [49, §52(a) p. 15].

[6]*Block productivity data* is a term used in Australia for data recorded at the mill for cane payment purposes [116]. The term *field records* is used in this dissertation to mean existing information on field productivity history.

[7]Lodged cane is a term for sugarcane that is not erect, but still alive and growing. Lodging may *e.g.* be the result of strong winds or weakness or overweight of the cane stalks.

[8]The canopy of sugarcane is its leaves. Singels *et al.* [187] uses FI_{PAR}, an acronym for *fraction intercepted photosynthetic active radiation*, to define the sugarcane canopy's ability to absorb light.

days after germination depends on temperature, crop class and varietal differences. Starting in December, one of the varieties required approximately 130 days to develop a full canopy as a first ratoon[9] and 160 days as a plant crop. Another variety used 120 days to develop an 80 % canopy for the first ratoon crop and 160 days for the plant crop. Starting in June, the varieties respectively used 170 and 200 days to develop an 80 % canopy and both varieties took nearly 300 days to develop a full canopy. However, Singels *et al.* concluded that these initial differences in growth diminish to the point of negligibility—in terms of final yield—as crops near the time of harvest, especially if a full canopy is developed long before the time of harvest. To put this into perspective, the planned time of harvest is seldom at an age of less than 365–670 days, while a full canopy is usually developed before the age of 200–300 days. Another conclusion that Singels *et al.* drew from this result is that row-spacing optimisation is only interesting for crops that are cut every 12 months and are under irrigation, due to the fact that canopy development compensates for any space differences between rows rather rapidly.

2.6 Deterioration models for damaged sugarcane

A significant part of the value of any sugarcane crop is the degree to which it has withstood the impact of extraneous events during its growth. Various events cause different degrees of damage to cane, and damage is at times best remedied by immediate harvesting while at other times the effects are slow to materialise or of little consequence. Cane is adversely affected by, for example, frost, long-term cool weather, strong winds, hail, bush pigs, warthogs, stalk borers, fires, drought, flowering, red rot and brown rust. Cane is sometimes damaged in densely populated areas, often due to carelessness with regards to fire. Some of these problems are reviewed and discussed in this section.

2.6.1 Harvest and fire

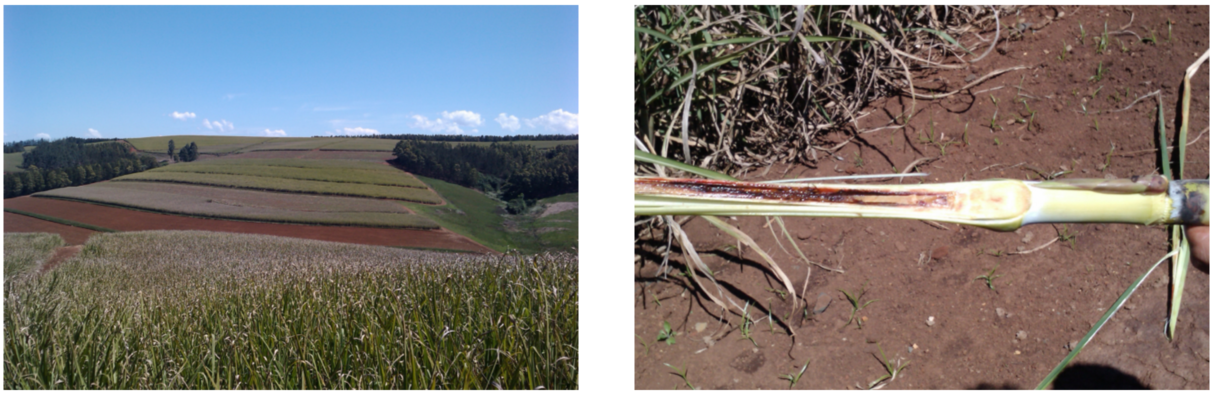
Wood *et al.* [236] analysed five trial plots in South Africa in 1972 designed for the purpose of determining the post-harvest deterioration of sugarcane, until 21 days after being cut, burnt and cut, or just burnt and left standing. Unburnt sugarcane lost approximately 17 % of its *recoverable sugar*[10] during the first 7 days and 38 % of its recoverable sugar after 21 days. Sugarcane that was cut immediately after being burnt, lost respectively 11 % and 53 % of its recoverable sugar during the 7 and 21 day periods. Burnt sugarcane that was left standing lost respectively 21 % and 57 % during the same periods. Wood *et al.* claimed that ambient weather is an important factor in determining the rate of deterioration following a fire.

In trials during 1973 designed to test for deterioration differences between varieties, Wood [234] found that variety was indeed a factor. It was also found that 24-month old cane deteriorated by approximately 4.5 unit % recoverable sugar per day, whereas 12- and 18-month old cane deteriorated by approximately 2.25 unit % per day (judging from [234, Figure 3, p. 138]).

Lionnet [142] concluded, based on trials, that trashing (cutting without burning) or burning and cutting are important factors influencing the deterioration of sugarcane, but that ambient temperature following the fire is even more important.

[9]A crop of crop class 1 is termed a *first ratoon* and a crop harvested for the third time is termed a *second ratoon*, and so on.

[10]*Recoverable sugar*, also known as *estimable recoverable crystal* or ERC, is the part of the sugarcane that is possible to recover as sugar [182].



(a) Frosted fields. (b) Dead growing point.

Figure 2.1: *Frost can kill sugarcane, but may sometimes only cause minor damage.*

Wood [235] showed in 1976 that billeted[11] cane deteriorates faster than whole stalk cane, especially for billet lengths under 20 centimetres (cm). In cane that was not burnt prior to harvest, whole stalk, 40 cm billets and 20 cm billets lost, on average, approximately 5.9 %, 6.1 % and 8.4 % of the available recoverable sugar in 5 days after being cut, respectively. The average effect was slightly greater for burnt cane, where whole stalk, 40 cm billets and 20 cm billets lost approximately 6.6 %, 8.3 % and 10.5 %, respectively. The worst case observed was for a 20 cm billets sample taken after 4 days whose recoverable sugar amount had decreased by 28 %.

In a trial in 1978, Clowes and Wood [38] tested the effect of chemical ripening on the deterioration rate of unburnt cane where unburnt, harvested, whole stalk cane in one of the trials lost approximately 17 % of the available recoverable sugar in 6 days and 31 % in 9 days, while in another trial it lost 34 % in 6 days and 41 % in 9 days. Clowes and Wood furthermore found that the ripener MON 8000 by Monsanto Chemicals may have the ability to slow down deterioration.

2.6.2 Frost

Frost can affect cane differently in terms of severity, ranging from mild effects to cracking of the stalks, in which case the cane dies and deteriorates rapidly. Figure 2.1 shows the characteristic greying of the tops which occurs when cane has suffered frost. The figure also shows a frost-damaged stalk, in which the growing point has died and side-shoots have begun to emerge.

During an experiment in Louisiana in 2001, Eggleston and Legendre [53] noted that at $-4.4°C$ lateral buds were damaged and at $-5.6°C$ there was cracking of the stalks. In their paper, Eggleston and Legendre [53, Figure 2, p. 454] present a figure in which a slow—approximately 1.5 parts per million dextran[12] accumulation in brix[13] in 14 days—deterioration occured as a result of the first $-4.4°C$ frost and a fast—approximately 10 parts per million additional dextran accumulation in brix in another 14 days—deterioration occured as a result of the $-5.6°C$ frost. The damage from frost at the mentioned levels is mainly due to secondary dextran production by *Leuconstoc* bacteria which invade the cracked and broken tissue.

Legendre *et al.* [136] studied the effect of a Louisiana frost event that took place in 2006 and

[11]Billeting is the act of cutting cane stalks into shorter pieces—called billets—usually for the purpose of making the cane conveyable through the interior of (mechanised) combine-harvesters.

[12]Dextran is a chemical which accumulates in sugarcane as it deteriorates.

[13]Brix is a term used in the sugar industry for all soluble substances present in sugarcane.

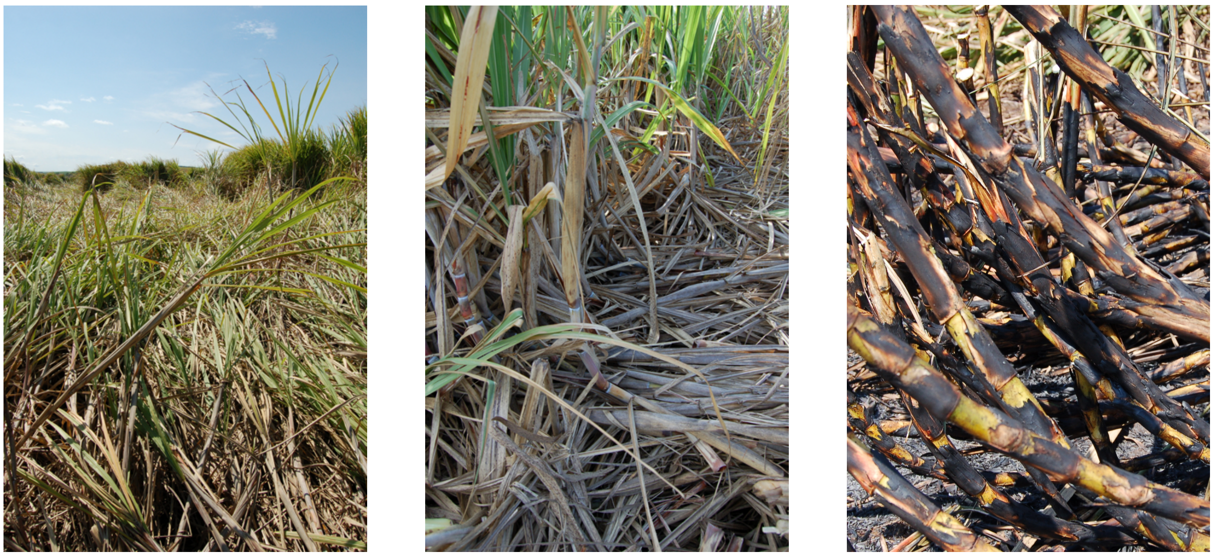

(a) Recently lodged cane. (b) Cane several months after lodging. (c) Burnt, lodged cane.

Figure 2.2: *Cane that falls or is blown down is termed lodged cane. Difficulties arise when the cane is allowed to grow for too long, since the stalks form a curve with ensuing handling problems.*

affected 30–40 % of the harvest. Temperatures between 0 and −2.2°C cosmetically affected leaves and buds while temperatures between −2.2 and −3°C killed the growing point and the top third of the stalk. The whole stalk was killed by temperatures below −3.9°C, while temperatures below −5.6°C usually caused cracking of the stalk rind. In terms of sucrose losses measured there were no significant losses at one of the sites that had experienced −1.7°C temperatures.

Another study in the United States of America conducted in Florida in 1979 by Gascho and Miller [65] focused on the effect on six different varieties of a series of closely intertimed frosts of which the coldest was −6°C. All varieties demonstrated a linear decrease over the period of nine weeks following the three day frost period, estimated at approximately 9.2 % sugar loss per week.

A study by Irvine [113] of a 1966 freeze in Louisiana concluded that there was a 0.014 % per day sucrose decrease during the 92 day period following a spell of −4.4°C temperatures and that, in fact, purity increased significantly during that period. Irvine concluded that cane with little stalk damage (top 1–2 inter-nodes) may be harvested as late as 3 months after the event.

2.6.3 Lodging

Strong winds combined with heavy stalks regularly leads to cane falling over, a phenomenon known as *lodging*. Most of the stalks survive and begin to grow vertically and after approximately three months, a complete quarter-circle has formed. Figures 2.2(a) and 2.2(b) respectively show cane that has recently lodged and cane that was lodged approximately two to three months ago. Notice the curved shape of the stalks in Figures 2.2(b) and 2.2(c). Cane does not bend immediately when lodging occurs; it is the subsequent months of growth that bends the cane.

Methods to predict and control lodging of sugarcane have been researched by Berding *et al.* [14]. In a large sampling experiment in Australia they quantified, with a 0.01 probability of being wrong[14], losses in brix from 15.162 % to 14.134 % and CCS from 10.323 % to 9.252 % between erect and lodged sugarcane respectively during pre-harvest months [14, p. 270]. Combined with the loss of stalk soundness, lodging caused 24 % losses in CCS (approximately 2.4 unit %). During harvest months, the losses in brix were on average 1.925 unit % and losses in CCS were 2.225 unit % [14, p. 271].

As noted by Berding *et al.* [14], Singh *et al.* [188] quantified the loss of CCS due to lodging to roughly between 3 % and 12 % which corresponds to a decrease from 10 unit % to between 9.97 unit % and 8.8 unit %, thus not contradicting the results of Berding *et al.* Furthermore, Singh *et al.* estimated cane yield losses to 11–15 % compared to same-aged erect cane.

Experiments by Sinclair *et al.* [186, Figures 5 and 6, p. 213–214] show that lodging may reduce the average inter-node volume[15] by approximately 0–25 % in stalks that have reached a vertical top-pose[16]. The volume loss may significantly differ between varieties.

During the 1990s, Seeruttun *et al.* [184] performed an experiment in Mauritius to quantify the cane and sugar yield difference between 12-month and 24-month old cane. The 24-month old crops produced 22–60 % less sugar than two 12-month crops, and Seeruttun *et al.* additionally argued that lodging increases weed-, pest- and harvesting problems.

2.6.4 Flowering

Flowering is the means by which sugarcane reproduces in the wild. Berding *et al.* [14] found that excessive flowering should be attributed with a 10 % (ca 1.5 unit %) sucrose loss in Australia. The applicability of this value in the South African case is unknown.

According to Nuss [162], a study performed in South Africa during the 1980s on flowered stalks during the period May–December showed that yield in terms of sucrose tonnage generally increases for the May–October period and that it may decrease during the September–December period.

In India, Rao and Kumar [175] experimentally ascertained that flowering stalks display a loss in cane yield of (judging from [175, Figure 1, p. 186]) approximately 10 % three months after flowering. Rao and Kumar argued, however, that sucrose levels actually increase—rather than decrease—due to flowering. The loss in sucrose recorded by Berding *et al.* should perhaps be construed as a result of sucrose weight-loss mainly due to stalk weight-loss. According to Rao and Kumar, flowering does not affect fibre content.

2.6.5 Eldana infestation

According to Leslie [138], Eldana[17] levels measured as the percentage of *inter-nodes*[18] with Eldana bore-holes (% IB) increase linearly with time at a rate of approximately 3 unit % in 4 months, which is equal to 0.75 unit % IB per month. The sugar yield losses due to the 3 %

[14]In statistical inference, the probability of being wrong when rejecting the null hypothesis is also known as the p-value.

[15]An inter-node volume was calculated by measuring the length of an inter-node and multiplying that by its mid-point cross section area.

[16]When the top part of the sugarcane stalk points towards zenith the cane is said to have reached a vertical top-pose.

[17]Eldana is a moth known to be a *stalk borer* which causes serious damage to sugarcane.

[18]An inter-node is a segment of a sugarcane stalk and begins and ends with a node—nodes are the points along the stalk from where shoots may germinate.

IB increase corresponded to approximately 1.8 % recoverable sugar. Thus, the increase in IB % per month is approximately equivalent to a decrease in RV % of $1.8/4 = 0.45$ unit %. The loss is thus $1.8/3 = 0.6$ % recoverable sugar per % IB. In a 100 t.ha^{-1} crop and with a 2000 Rand per tonne (R.t^{-1}) RV-price, the monthly IB % increase is approximately equivalent to $0.0045 \times 100 \times 2000 = 900$ R.ha^{-1} in lost profit. By means of pesticides, the IB % increase may be reduced by approximately 40–75 % [76, 138].

The linear increase of % IB reported in [137] may, according to a study by Carnegie and Smaill [28] in South Africa in 1982, be decreased through a round of *de-trashing*[19] by 5–10 % in four months.

In trials by Goebel and Way [76] performed in Zululand (South Africa) during the period 2001 to 2002, results showed that in Gingindlovu (one of two sites) the decrease in recoverable sugar percentage per % IB is approximately 0.02 % recoverable sugar per % IB on the % IB interval 2.1–6.2. On the % IB interval 6.2–18.4, decrease in recoverable sugar percentage per % IB is approximately 0.15 % recoverable sugar per % IB. In Empangeni, the interval 1.1–2.3 showed a 0.08 % recoverable sugar per % IB decrease, while a 2.3–17.4 interval produced a value of 0.13 % recoverable sugar per % IB decrease. These values differ from those of [138].

A survey of available data throughout South Africa by Goebel *et al.* [77, Figure 6, p. 344] in 2003 indicated that Eldana levels increase by 1.3 unit % *stalks bored* (SB) per month in Darnall and Entumeni, but by approximately 2.7 unit % SB in the Amatikulu mill region. The so-called *economic injury level* reported by Goebel *et al.* is approximately equal to that of Leslie [137], namely 7 % IB[20]. Goebel *et al.* [77] provide the conversion rate of 1 : 7.7 for SB[21] to IB %, which leads to the deduction that Eldana infestation levels increase by 0.17 unit % IB per month at Darnall and Entumeni to 0.35 unit % per month at Amatikulu. These values are slightly smaller than those deduced from [137] (0.75 unit % per month). The conversion rate from SB to IB is, however, closer to 1 : 6, judging by the results of Cadet *et al.* [26, Figure 3, p. 532], which would raise the respective values of % IB increase per month from 0.17 and 0.35 to 0.22 and 0.45 unit % IB per month.

According to Goebel *et al.* [77] as well as Keeping and Rutherford [119], susceptibility to Eldana is different among varieties and is correlated with inter-node rind hardness, and thus also correlated with fibre content, which is usually negatively correlated with recoverable sugar. This means that variety selection based on inter-node rind hardness to combat Eldana may be an undesirable alternative.

2.7 The SASRI information sheets

The foremost authority on sugarcane biology in South Africa is the South African Sugar Association's (SASA) Sugarcane Research Institute (SASRI[22]). SASRI is mandated to conduct research on sugarcane on a continual basis, and possesses a vast accumulated amount of information, data and knowledge. SASRI publishes a binder with informative leaflets known as *information sheets* [182] which summarise research results to date and provide instructions on how to deal

[19]De-trashing consists of removing dry leaf matter from sugarcane fields by pulling or beating the dead leaves off of the stalks.

[20]The measurement units used by Goebel *et al.* [77] is actually % stalk length red (SLR), which is a generalisation of % IB.

[21]Goebel *et al.* [77] used *stalks damaged* (SD), which is equivalent to *stalks bored* (SB).

[22]SASRI is an acronym for *South African Sugarcane Research Institute* and is based at Mount Edgecombe, KwaZulu-Natal.

with various issues that may occur at sugarcane farms. The information that bears implications on this dissertation is briefly summarised in the subsections below. The shaded conclusions with respect to *"impact on harvest scheduling"* following each paragraph are specifically for the South African case, deduced by the author of this dissertation based on the information sheets.

2.7.1 Commercial sugarcane varieties

The relevant properties of South Africa's commercial sugarcane varieties as presented in the information sheets are summarised in Table 2.1. Different commercial varieties display vastly different properties. Some varieties perform better in terms of RV % than others, but worse in terms of cane yield. Some varieties are especially suited for dry conditions, while others achieve low yield but high RV %, which makes them more suitable for growing far from the mill. Some varieties are neither high-yielding nor of high RV %, but are highly resistant to diseases. Each variety is useful under certain conditions, and careful consideration is required when selecting varieties for planting.

2.7.2 Common diseases

Sugarcane is constantly under risk of becoming infected by various diseases, and some of these diseases bear implications when considering which fields to harvest. Many of them may be combated by means of pesticides or good farming-practices. Outbreaks of these diseases have to be taken into account when constructing sugarcane harvesting schedules.

Brown rust

Brown rust is caused by the fungus *Puccinia melanocephala* which is favoured under cool, wet conditions with high humidity and below 30°C temperatures. Susceptible varieties may lose as much as 20–30 % in cane yield and younger cane (3–6 months) is more susceptible than older cane. The recommended control measures are to avoid planting susceptible varieties on south-facing slopes, in valley-bottoms or near tree-lines while ensuring good nutrient balance and considering that cane planted or harvested during spring or summer is less prone to infection due to its higher age during the wet season.

Brown rust impact on harvest scheduling: Infected N29 or N33 fields will experience decreasing cane yield. Such fields are, however, usually too young to harvest.

Mosaic

Mosaic is a viral disease spread by either infected seed cane *setts*[23] or the aphids[24] *Hysteroneura setariae* and *Rhopalosiphum maidis*. Effects on sucrose yield are reductions of 30–40 % by mass when all stalks are infected, for both varieties NCo376 and N12. Varieties differ in susceptibility. Recommended control actions are to plant less susceptible varieties, to ensure that seed cane setts are free of mosaic, to ensure thorough crop *eradication*[25], to avoid planting during mid-

[23]The pieces of stalk laid in furrows during planting are called *setts*.
[24]Aphids are various species of a family of flying insects.
[25]The process of ensuring that an old crop is completely destroyed, particularly with respect to its roots [182].

Name	Harv. time	Harv. age	Soil pot.	Cane yield	RV %	Flowering freq.	Lodging susc.	Smut susc.	Mos. susc.	RSD susc.	Rust susc.	Red rot susc.	Nem. susc.	Eld. susc.	Dry cond. grow.
NCo-376	Jul–Dec	12	mod–good	high	low	mod	low	high	very high	high	low	high	high	high	poor
CP66/1043	Apr–Jun	–	–	low	very high	low	mod–high	low	low	high	low	–	–	mod–high	poor
N12	May–Oct	16–22	poor	mod–high	mod	mod	low	mod	mod	mod–high	low	low	mod	low–mod	mod–good
N14	Jul–m Aug	12	mod–good	high	low	high	low	mod	mod	high	mod	low	low	high	poor
N16	Jul–Dec	C12 M18–24	good	high	mod	low	mod–high	high	mod	mod–high	mod	mod	high	high	mod
N17	Aug–Dec	C12	good	low–mod	mod–high	high	mod–high	low	mod	very high	low	high	high	low–mod	mod–good
N19	Apr–Aug	12	good	low–mod	high	low	early & high	low	high	high	low	high	very high	mod high	poor
N21	Aug–Dec	14–15	poor–good	high	mod	low	high	low	low	high	low	mod	very high	low	good
N22	Apr–Aug	–	good	low–mod	high	low	low	very low	low	mod	low	–	–	mod–high	poor
N23	Jul–m Aug	–	mod	mod	low	high	low	low	mod	mod–high	low	low	mod	mod	mod
N24	Apr–Oct	–	good	low	very high	low	high	low	low	high	low	high	very high	mod	poor

Table 2.1: *Selected properties of commercial South African sugarcane varieties NCo376–N50. The abbreviations are as follows: Harv. = Harvest, pot. = potential, freq. = frequency, susc. = susceptibility, Mos. = Mosaic virus, Nem. = Nematodes, Eld. = Eldana, cond. = conditions, grow. = growth, m = middle of, irrig = under irrigation, rainf = rain-fed, mod = moderate, M = midlands only (KwaZulu-Natal), C = coastal region only and "–" means that no information was available.*

Name	Harv. time	Harv. age	Soil pot.	Cane yield	RV %	Flowering freq.	Lodging susc.	Smut susc.	Mos. susc.	RSD susc.	Rust susc.	Red rot susc.	Nem. susc.	Eld. susc.	Dry cond. grow.
N25	Aug–Dec	–	poor	high	low	low	mod–high	mod	mod	very high	low	high	mod	low–mod	mod
N26	Apr–Jun	12	irrig good	low	very high	low	high	low	low	high	mod	low	very high	very high	poor
N27	Jul–m Oct	12	good	high	high	high	low	low	low	low–mod	mod	low	very high	high	good
N28	Jul–Oct	–	irrig poor–good	mod	mod	low	low	mod	mod	mod–high	low	mod	high	low–mod	poor
N29	Apr–m Oct	15	good	low	very high	high	mod	low	mod	high	high	low	high	low–mod	poor
N30	Apr–Jun	12	irrig good	low	very high	mod	mod–high	low	low	high	low	high	very high	very high	poor
N31	Jul–Dec	18–24	poor–mod	very high	low	mod	mod–high	high	mod	high	mod	high	mod	mod	good
N32	May–Oct	–	irrig poor–good	mod	high	mod	low	mod	high	high	mod	mod	mod	low–mod	poor
N33	Jul–Dec	15–18	poor	mod–high	mod	low	low	mod	mod	low–mod	high	–	mod	low	good
N35	Apr–Dec	12	good	low	high	low	low	low	low	mod	mod	mod	high	high	poor
N36	Apr–Nov	12	irrig mod–good	mod	high	mod	mod	mod	mod	mod–high	low	low	high	mod–high	mod

Table 2.1: *(continued)*

Name	Harv. time	Harv. age	Soil pot.	Cane yield	RV %	Flowering freq.	Lodging susc.	Smut susc.	Mos. susc.	RSD susc.	Rust susc.	Red rot susc.	Nem. susc.	Eld. susc.	Dry cond. grow.
N37	May–Dec	15–24	good	mod–high	high	low	mod	high	low	mod	mod	mod	high	mod	poor–mod
N39	Jul–Dec	15–24	poor	mod–high	high	mod	low	high	mod	low–mod	mod	–	mod	low	good
N40	May–Nov	12	irrig good	low	very high	mod	low	low	low	low–mod	low	–	mod	mod	poor–mod
N41	Apr–Oct	12–24	irrig rainf good	low–mod	high	low	mod	mod	mod	high	mod	–	mod	low–mod	mod–good
N42	Apr–m Oct	C12	mod–good	mod–high	high	high	low	low	low	–	mod	–	mod	low	–
N43	Aug–Dec	12	irrig good	mod–high	mod	low	mod	high	low–mod	–	low–mod	–	–	mod–high	poor
N44	Jul–Dec	M20–24	mod–good	high	mod	low	mod–high	high	low	–	low	–	–	mod–high	poor
N45	–	C12–14	mod–good	mod–high	mod	low	mod	mod–high	low	–	low	–	–	mod–high	–
N46	Aug–Dec	C12	irrig	mod	mod	low	mod	low–mod	low	–	mod	–	–	mod	poor
N47	–	15–18	poor–mod	mod	high	low	mod	mod	mod	–	low	–	–	low–mod	–
N48	–	M20–24	–	mod–high	high	very low	mod	mod	low	–	low–mod	–	–	mod	–
N49	Mar–Dec	C12	irrig	mod	mod–high	low	mod	low	low	–	mod	–	–	mod–high	–
N50	–	M20–24	mod–good	high	mod	low	mod–high	high	low	–	mod	–	–	mod	

Table 2.1: *(continued)*

October to late-January (so as to avoid having young crops during peak aphid activity periods) and to weed and rogue isolated fields that show low infection levels.

Mosaic impact on harvest scheduling: Isolated occurrence—none; Extensive occurrence—harvest, plough-out and plant with resistant variety.

Pineapple disease

Pineapple disease is caused by the fungus *Ceratocystis paradoxa* and can halt germination. Different varieties experience different susceptibility and environmental conditions play a part in mitigating or propagating the disease. Control measures include planting under non-slow germination conditions, moisture application to furrows after planting and the avoidance of excessively deep planting.

Pineapple disease impact on harvest scheduling: Harvest, plough-out and re-plant early during the year so that germination takes place at time of rapid germination.

Pokka boeng

Pokka boeng is spread by the wind-borne spores of the fungi *Fusarium moniliforme* and *F. subglutinans*, and may, in some varieties, lead to severe crop loss. The control actions against this disease are limited to the withdrawal of susceptible varieties by SASA.

Pokka boeng impact on harvest scheduling: None.

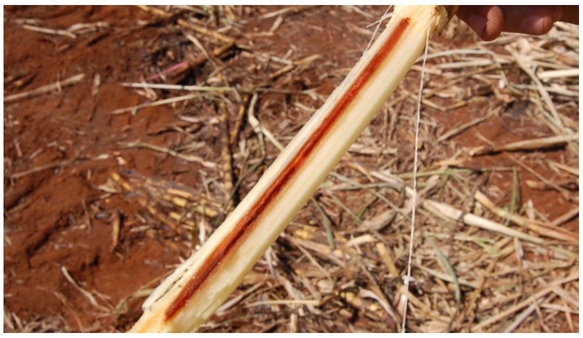
Ratoon stunting disease

Ratoon stunting disease (RSD) is an infection caused by the bacterium *Leifsonia (Clavibacter) xyli xyli*, which affects cane yield with a loss of up to approximately 40 %. Effects vary between varieties, between plant and ratoon crops and when comparing normal conditions to droughts. The recommended control actions are to ensure healthy seed cane setts, to harvest infected fields early, to eradicate the old crop thoroughly and let the fields lie fallow for three months during the winter, before being replanted.

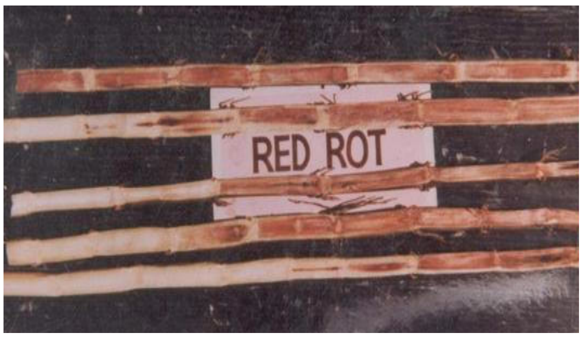
RSD impact on harvest scheduling: Harvest before June. Cane yield expectations should be decreased.

Red rot

A disease that sometimes strikes sugarcane is called *red rot* and is caused by the fungus *Colletotrichum falcatum* [223] (also known as *Glomerella tucumanensis*). The effects are severe [209]. The appearance of stalks having been damaged in other ways is sometimes mistaken for the disease. Stalk damage with ensuing rotting (not red rot) is shown in Figure 2.3(a) while red rot is shown in Figure 2.3(b).



(a) Damaged cane.



(b) Red rot. Shown with permission [200].

Figure 2.3: *Damage caused by borers or other mechanical damage may sometimes be mistaken for* Colletotrichum falcatum, *all three constituting problems facing sugarcane growers before the cane may be harvested.*

Red rot of sugarcane leads to sucrose loss in infected stalks, amounting to 20 % or more in individual fields. Old, frost-, drought- or borer damaged cane in cooler areas is more susceptible to red rot. To control red rot, highly susceptible varieties, unhealthy seed cane and allowing cane to grow too old should be avoided. Borer damage should be controlled and fields affected by red rot should be harvested early.

Red rot impact on harvest scheduling: Harvest soon due to deterioration of sucrose levels.

Smut

Smut is a disease that spreads through wind-borne spores of the fungus *Ustilago scitaminea.* Yield losses caused by smut generally increase with increased crop class.

Smut impact on harvest scheduling: Severe cases—harvest and plough-out before end of June. Plant resistant variety.

Sour rot

Sour rot is a disease that spreads through wind-borne and rain-splashed spores of the fungus *Phaecytostroma sacchari.* Yield losses caused by this fungus may become severe during drought.

Sour rot impact on harvest scheduling: Severe cases—harvest as soon as possible and avoid allowing crops to over-mature.

2.7.3 Environmental and biological events

Utilising and exploiting the prevailing environmental conditions is a precursor to success in sugarcane farming. Sugarcane needs sunshine, water, nutrients, heat and shelter from adverse conditions. Shelter or protection in the form of available insecticides are not always available, with the result that sugarcane suffers, at times, from accidental fires, drought, frost, flowering of the cane, borer infestations, aphid infestations and green leaf sucker infestations. These environmental conditions, extraneous events and insect-related problems also have to be taken into account when constructing a sugarcane harvesting schedule in practice, and the SASRI information sheets contain advice with respect to these various events.

Accidental fire

A field subjected to fire by accident may be categorised as subject to one of four different recommended actions. Firstly, if the fire was relatively cool, the cane may be left, at no great loss, in the field to recover and continue to grow normally. Secondly, if a sample of stalks shows that the *meristem*[26] is dead, the fire was warm and the cane has visible but unmillable stalks, it should be cut back and treated as normal re-growth, unless in winter, in which case it should only be cut once the rains have set in. Thirdly, if the fire was warm and the cane stalks are invisible, the cane should be left in the field to tiller[27]. Finally, if the cane is millable it should be harvested immediately since it deteriorates by 1–2 unit % recoverable sugar per day after being cut and even faster if left standing in the field.

Accidental fire impact on harvest scheduling: warm fire—harvest immediately unless in winter, in which case harvest when the rains have set in; cool fire—leave and harvest according to the original plan, expecting slightly less cane yield.

Drought

Drought-stressed cane should be harvested if the stalks are millable; Eldana-prone fields should receive first priority, drought sensitive varieties second priority and cane on shallow soil or that face in a northerly direction should receive third priority. Fields with unmillable cane should be cut back if Eldana hazard levels are high or should else be left standing if the crop can survive. If the crop is unmillable and cannot survive, it should be cut back at the onset of the rains.

Drought impact on harvest scheduling: Eldana-prone cane should be depreciated heavily, more so for drought-sensitive varieties and dry field situations (hill-tops, shallow soil or north-facing slopes).

Frost

If sugarcane is subject to -2°C, there is slight damage to the meristem, striping of leaves, tip die-back and a browning of the leaf roll above the meristem, but negligible hampering of growth. At -3°C, there is partial damage to the meristem, with die-back of leaves, but re-growth occurs within a few weeks. At -5°C, the meristem is killed, a few buds below the *natural breaking point*[28] die, 40 % of the leaves die, growth is halted, but the cane may be left standing for approximately three weeks. At -7°C, the top six buds below the breaking point are killed, all green leaves are killed—which leads to the cessation of photosynthesis in the affected stalks, and since RV % decreases rapidly, the cane should—if it is millable—be harvested without undue delay. At -9°C, most buds are killed, inter-nodes turn glossy and deterioration is extremely

[26]The meristem is the part of the stalk that contains cells which are not completely distinguishable from each other and constitutes the point of growth.

[27]Develop lateral shoots from the base of the stalk.

[28]A point on the stalk below the meristem at which it breaks easily after removing the sheaths covering the stalk's upper parts [111].

rapid. Harvesting should commence immediately. If the weather is dry as well as cold, a so-called *black frost* may occur. A black frost may freeze the sap in the plants, which leads to rapid deterioration and it may strike a very large area on a single occasion.

Frost impact on harvest scheduling: To compensate for the effect of the various levels of frost, there should be no adjustment of yield for the $-2°C$ level, a slight negative alteration of the yield expectations for a $-3°C$ level frost, a halting of the yield development starting immediately combined with a depreciation of RV % starting three weeks after a $-5°C$ frost, a strong depreciation of RV % directly following a $-7°C$ frost and the same for a $-9°C$ frost or a black frost.

Flowering

The flowering of sugarcane is its natural means of reproduction and extensive instances—more than 20 % flowering stalks—may impact RV % and yield. Approximately four months after the onset of flowering, or if entering October, sucrose levels may begin to decrease. Before that, sucrose levels are usually higher in flowered stalks than in other stalks.

Flowering impact on harvest scheduling: If flowering is very extensive, a slight depreciation of RV % after four months, or after the beginning of October, should be applied.

2.7.4 Insects

As with diseases and extraneous events, insects may become a pest with respect to sugarcane production. They may use the plant for feeding or other activities, causing serious damage, and invasions of sugarcane by insects are therefore also an important factor to consider when constructing a sugarcane harvesting schedule.

Eldana borer

The moth *Eldana saccharina*, mentioned in §2.6.5, lives for approximately one week, during which it lays eggs in concealed dry-matter parts of sugarcane. The eggs hatch within 7–10 days, and the larvae disperse to mine sugarcane for 20–60 days before they pupate for seven days, whereupon the insect begins its next life cycle. Eldana larvae bore holes in the stalks and feed off the matter inside; the bored inter-nodes are then often invaded by red rot. Damage may become severe, especially in moisture stressed crops.

Eldana impact on harvest scheduling: Eldana infested fields should be harvested within 20 days or as soon as pupae appear, in order to prevent the completion of a life cycle. During the 20 days and for the remainder of the season, a depreciation of yield should be applied.

Sesamia borer

The moth *Sesamia calamistis* causes the same type of damage as Eldana, but in younger cane and at a lower rate and intensity. The cane is often young enough to be able to grow new tillers subsequent to a Sesamia borer attack. However, since Sesamia infestations seem to promote Eldana infestations, early harvest of such cane is recommended.

Sesamia impact on harvest scheduling: Harvest within 20 days. After that, an eldana-risk related depreciation of the cane yield should be applied.

Chilo borer

Chilo sacchariphagus is also a stalk borer and is a serious pest in the Far East, Mauritius and Madagascar. As opposed to Eldana, the Chilo borer prefers to lay its eggs on the green matter, rather than the dry matter. The larvae then begin to feed inside the upper parts of the stalk, pupating inside, completing a life cycle in approximately 60–70 days. This pest may be controlled by biological means, and no implications on harvest scheduling are known.

Chilo impact on harvest scheduling: None.

Hysteroneura

The aphid *Hysteroneura setariae*, mentioned earlier in this section, should be controlled by avoiding planting during the summer.

Hysteroneura impact on harvest scheduling: None.

Green leaf sucker

The 7 millimetre long insect *Numicia viridis* is known to poison sugarcane and can, in extreme cases, cause severe yield and sucrose loss.

Green leaf sucker impact on harvest scheduling: If the outbreak is severe, depreciate cane yield and RV % slightly over time.

2.8 Chapter summary

In this chapter, the various approaches taken in the literature with respect to modelling sugarcane have been reviewed briefly. Several mixed integer linear programming models as well as discrete event simulation models were found, and the problems modelled range from highly delimited to very broadly defined scenario-type exploratory questions. In Australia, quite a few models have reached the stage of providing actual decision support, and some of them are integer programming-based and some are simulation-based. There are many models, but few OR tools of validated quality available to growers for seasonal harvest scheduling.

Further on in the chapter, a survey of the current issues pursued within the field of supply chain management is followed by a brief review of current agricultural value chain research. Within the field of SCM-related research, theories are still under development. The scientific community does not yet agree on exactly how modelling in terms of, for example, performance measurement of supply chains should be undertaken in order to provide successful SCM-improvement projects.

Various modelling aspects of sugarcane growth and deterioration were also reviewed and discussed in this chapter, and most events that influence the growth or deterioration seem to have been previously studied and quantified to some extent. There is, however, a lack of data that may be used for computing accurate and precise deterioration functions. If harvest scheduling decision support is to become an accurate science that can give precise advice, more work must be done on quantifying the deterioration rates of sugarcane for various varieties, diseases, field conditions and all other factors involved. Research on post-harvest deterioration in [38,234–236], where deterioration rates were at the centre of the study, set good examples of what is needed for the other events.

This chapter is presented in partial fulfilment of Dissertation Objectives I, III and IV, presented in §1.3.

CHAPTER 3

The sugarcane supply chain

Contents

This chapter contains a brief description of the main activities occurring in the sugarcane supply chain, prior to the cane reaching the crusher-stage of the sugarcane milling process. South African regulations pertaining to sugarcane production are mentioned and the payment system is briefly explained.

3.1 Introduction

Global sugar production is currently estimated at 160 million tonnes per year, according to the United States Department of Agriculture [217]. The refined product, *centrifugal sugar*, is mainly produced from sugarcane or sugar beets, and for every tonne of sugar that is derived from sugar beets, approximately three tonnes are derived from sugarcane. Sugar beets are mainly grown in temperate climates in the northern hemisphere, while sugarcane is grown in sub-tropical and tropical climates in both the northern and southern hemispheres. During the 2008/2009 sugar season, Brazil is estimated to have been the largest producer with its 33.7 million tonnes, of which 21.6 million tonnes was exported. India was the second largest producer with 24.8 million tonnes, dropping sharply from 28.9 million tonnes during the 2007/2008 season. India decreased its exports from 2.4 million tonnes during the 2007/2008 season to 1.3 million during the 2008/2009 season. The European Union (EU) produced 16.8 million tonnes and exported 1.4 million tonnes during the 2008/2009 season. China produced 14.4 million tonnes and imported more than it exported, while Thailand produced 7.2 million tonnes of which 5.6 million tonnes was exported, which makes it the second largest exporter of centrifugal sugar in the world. Australia produced 5.2 million tonnes and exported 3.9 million tonnes of centrifugal sugar during the 2008/2009 season.

South Africa is currently the 13th largest producer at 2.3 million tonnes of centrifugal sugar and exports 1 million tonnes per year. The United States of America produces 7.4 million tonnes, Mexico 5.9 million tonnes, Pakistan 3.7 million tonnes, the Russian Federation 3.1 million tonnes, Argentina 2.5 million tonnes and Colombia 2.4 million tonnes.

3.2 The South African sugar industry

South African commercial sugar is exclusively produced from sugarcane, grown, as mentioned in §1.1, in the provinces of Mpumalanga and KwaZulu-Natal. There are fourteen sugar mills in the country, operated by six different companies, namely Gledhow Sugar Company Ltd (GS), Illovo Sugar Ltd (Il), Tongaat Hulett Sugar (TH), Transvaal Sugar Ltd (TSL), Umfolozi Sugar Mill Ltd (USM) and Union Cooperative Ltd (UCL). The fourteen mills are (ownership in paranthesis) Malelane (TSL), Komati (TSL), Pongola (TSL), Umfolozi (USM), Felixton (TH), Amatikulu (TH), Darnall (TH), Gledhow (GS), Maidstone (TH), Noodsberg (Il), Union Coop (UCL), Eston (Il), Sezela (Il) and Umzimkulu (Il). Malelane and Komati are situated in Mpumalanga, while the other twelve mills are all in Kwa-Zulu Natal, as shown in Figure 3.1. During 2008, the mills crushed between 920 000 and 2.2 million tonnes of sugarcane each.

The South African sugar industry is regulated by means of the Sugar Act (Act No 9 of 1978) and its amendments. The act institutes SASA and its constitution, and provides for the Sugar Industry Agreement (SIA) [49] which, upon ministerial agreement, dictates how Mill Group Boards and other industry representative bodies should function and regulates the relationship between millers and growers. The SIA is a document that contains rules regarding the practical aspects of the South African sugarcane supply chain and it was revised in 2000, mostly towards deregulating the industry [48]. SASA conducts a substantial amount of research on sugarcane through its subsidiary, SASRI.

The smallest sugarcane growers in South Africa grow sugarcane on an average of 1.7 hectares of land [194] and they are usually *grouped* together in one of several ways. They may have signed a contract with a company, usually referred to as a *contractor*, to transport their cane from the field to the mill. The contractor acts as a middle man between growers and a mill, and usually controls when each grower harvests. In practice, the miller assigns each grower a daily rateable delivery allocation based on the grower's estimated cane yield, the mill performance, the number of milling days per week and the length of the milling season. The growers then surrender their allocation to the contractor who, in turn, pools all his growers' allocations. This is known and endorsed by the miller, who then accepts harvested cane from the contractor based on the pooled allocation. However, some small-scale growers form formal or informal cooperatives and aquire the vehicles and equipment that are necessary to perform their own harvesting, loading and transportation of cane. Other small scale growers do not employ a contractor or establish relationships with other growers, but rather deliver their cane directly to the mill when they see fit. The small amounts that they deliver are often not regarded by the miller as disruptive to the consistent flow of cane sought [47]. To put these growers' impact on the supply chain into perspective, it may be mentioned that 1.7 hectares yields in the order of 100 tonnes of cane per year, which is comparable to what commercial growers deliver to the mill in a single day. Even so, if the number of small-scale growers is substantial, the small-scale growers may cause congestion at the mill due to their erratic delivery pattern [68]. Growers prefer to harvest early in the morning and during the part of the season when the cane is of the highest quality. Again, the fact that these growers often do not own their own long-range transport vehicles, may balance out the problem of too many growers preferring to harvest at the same time.

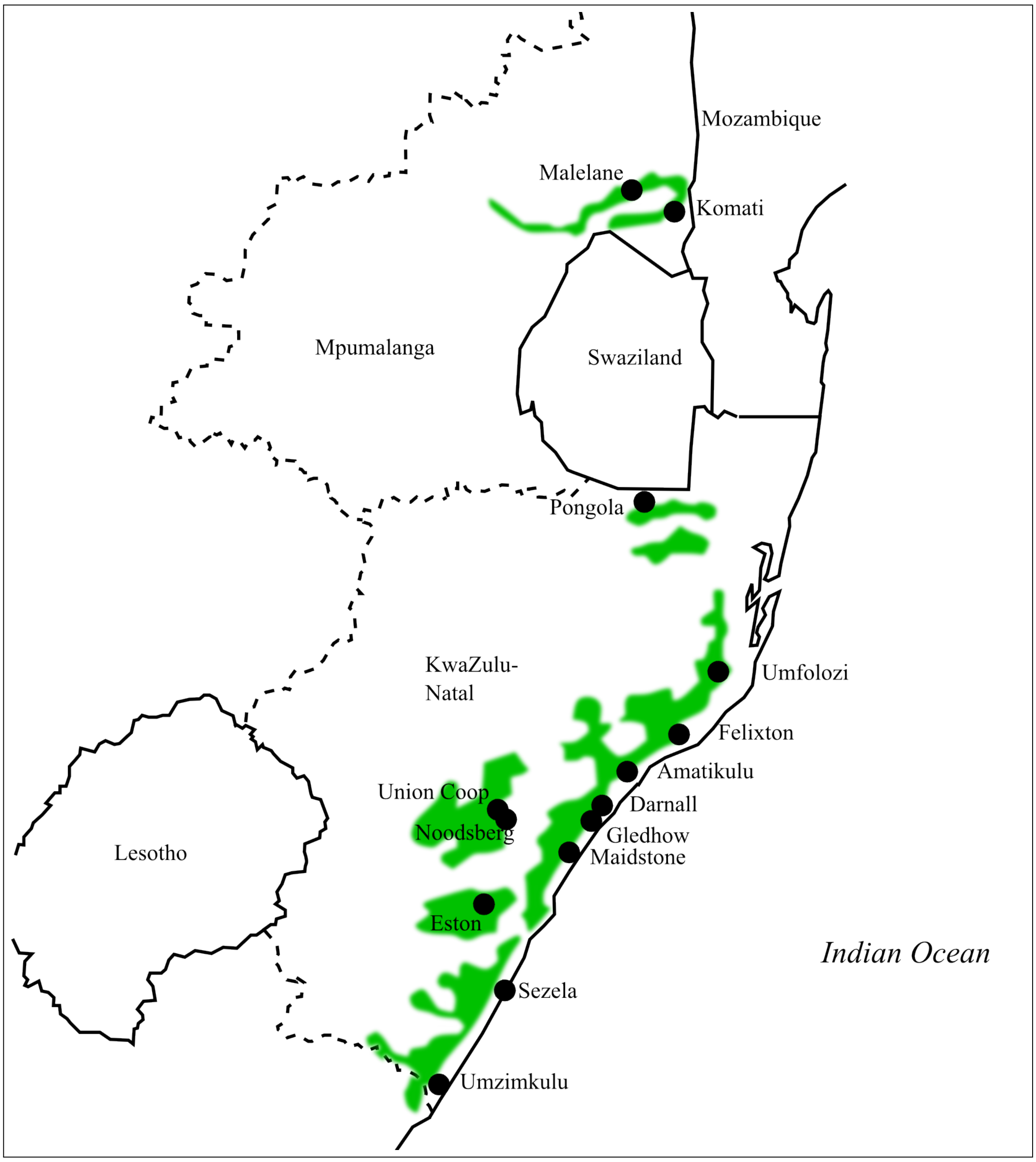


Figure 3.1: *Sugar mills and milling regions in South Africa. Map drawn using data from [182, 202].*

Family-owned sugar growers average 186 hectares of sugarcane fields which corresponds to approximately 10 000 tonnes of cane per year [194]. The SIA in effect makes the issue of timing and mode of delivery of harvested cane from these growers to the mill a Mill Group Board responsibility. This responsibility has in practice developed into the contractual concept of the DRD. The DRD is the amount (measured in tonnes of cane) that a grower is obligated to deliver to his or her crush[1] mill on a daily basis. The amount depends on the number of days per week that the mill is open to receive cane deliveries, denoted by t^{crush}, the current mill crushing

[1]According to the SIA, the home mill, or crush mill, is the mill to which the grower is bound by contract to deliver cane.

rate in tonnes per week, denoted by r^{crush}, the total cane in tonnes to be delivered during the remainder of the season by the grower, denoted by m^{grower}, and the total cane in tonnes to be delivered during the remainder of the season by all growers, denoted by m^{all}. The DRD, denoted here by D, may be expressed as

$$D = \frac{m^{\text{grower}}}{m^{\text{all}}} \cdot \frac{r^{\text{crush}}}{t^{\text{crush}}}. \tag{3.1}$$

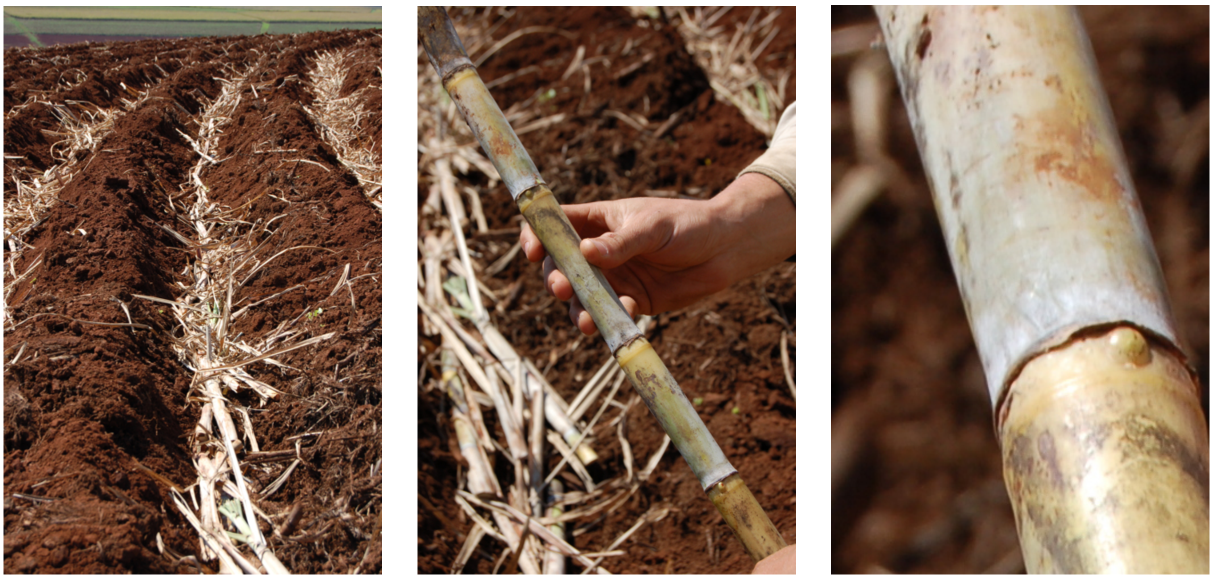
Here $m^{\text{grower}}/m^{\text{all}}$ is the fraction of all cane in the mill area that the grower in question accounts for. Furthermore, $r^{\text{crush}}/t^{\text{crush}}$ is the total daily tonnage to be crushed by the mill. The expression in (3.1) represents the rule according to which all growers should deliver cane consistently. The effectiveness of this system of daily rateable deliveries may, however, be questioned. The inherent variability in agricultural activities has caused the SIA and the Mill Group Boards to render adherence to the daily rateable delivery subject to a certain degree of flexibility. The flexibility itself does not appear to cause a problem, because the average of all exercised flexibility tends to zero as the number of growers increases. However, problems arise when growers over-estimate their total seasonal cane yield in order to be allocated a higher DRD so as to increase the probability of completing their harvesting operation before the rains set in [47]. It may be problematic (in a fairness sense) when a miller requests growers to deliver above the original daily rateable delivery, because the mill has managed to reach a higher crush rate than the original r^{crush}, and then unjustly reduces the daily rateable delivery at a later date, not returning the favour to the growers (of allowing those growers who answered to the request the same flexibility) [164].

Another problem related to queueing at the mill-yard arises and has been analysed by means of advanced tools such as discrete event simulation [87, 106]. The problem occurs at mills that do not schedule the cane deliveries consistently throughout the day, due to growers preferring to send cane early in the morning and on particular days of the week. One solution, according to [68], is to implement computerised and GPS-based vehicle delivery scheduling systems.

3.3 Sugarcane cultivation

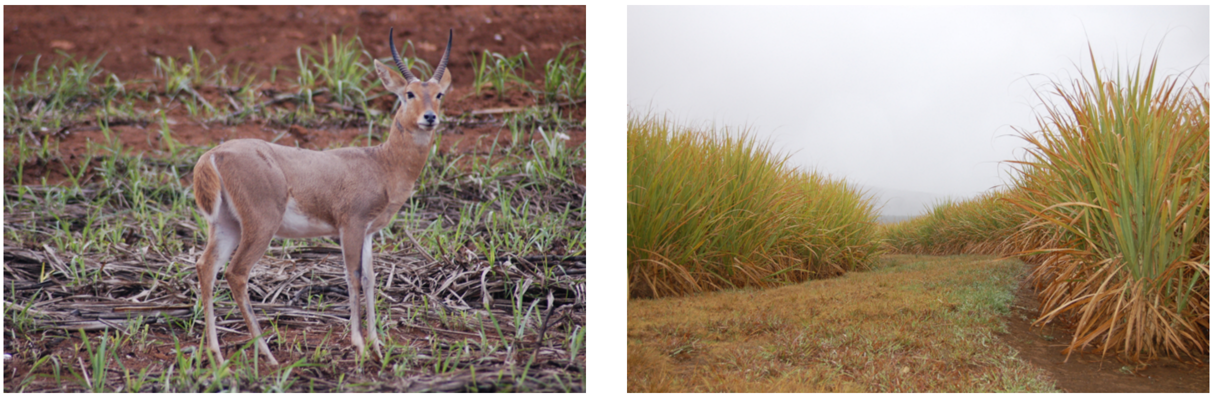
Sugarcane is preferably ploughed out during the first half of the harvest season, since there may otherwise not be enough time for planting and germination before the summer, during which growth is the fastest. Planting consists of placing seed cane setts one after the other in furrows created by a plough (also called a ridger). Furrows and seed cane are shown in Figures 3.2(a), 3.2(b) and 3.2(c). The new shoots germinate from buds located between the inter-nodes of the seed cane stalk.

Sugarcane is allowed to grow for a period of nine to approximately twenty-four months, depending on various agroclimatic factors. Close to the equator, temperature, radiation units (light) and precipitation combine to form conditions that allow sugarcane to grow to maturity within approximately nine months. In South Africa, cane is usually harvested at an age of between twelve to twenty-two months; a harvesting age of twelve months is common in coastal regions while an age of twenty-two months is more common inland [164]. Figures 3.3(a) and 3.3(b) show cane at ages one and six months, respectively, while Figures 3.4(a) and 3.4(b) show cane above twenty months of age. Figure 3.5 shows the age-mosaic character of a well-managed sugarcane farm mentioned in §1.2. SASRI has developed the *Farming Calendar* to aid growers in organising the various activities required for successful farming throughout the year [181].



(a) Furrows with seed cane. (b) Seed cane sett. (c) A Bud.

Figure 3.2: *Furrows and cane setts. Buds are visible between the inter-nodes.*



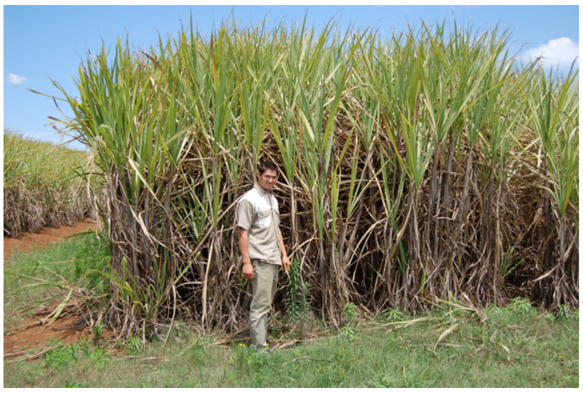
(a) New cane. (b) Young cane.

Figure 3.3: *Cane at ages one and six months.*

Different cane varieties exhibit different strengths and weaknesses in terms of resisting adverse conditions. In Figures 3.6(a) and 3.6(b) the difference between the height of the *millable*[2] part of the stalk of the cane variety N12 is compared to the N37-variety. The N37 stalks were approximately 40 % longer and slightly thicker than their same-age N12 counterparts. The previous 20 months had been relatively cold [164].

Chemical agents whose effects include a temporary stressing of the sugarcane plant are applied to some of the crops 6-12 weeks before harvest, thereby increasing their sucrose content. The procedure is called *chemical ripening* and is used around the world, the effect of which varies with cane variety, climate and chemical ripener [192]. Chemical ripening initiates behaviour of the cane identical to that which is induced by drought [112]. In South Africa, the ripeners include ethephon, Fusilade super, Gallant super, glyphosate, MON 8000 and Polaris CP 41945 [192] and at their best, the documented effects of these chemicals yield approximately 1.0 to 2.0 unit percent increase in recoverable sugar. A chemically ripened field should not be harvested before

[2]The millable part of the stalk is the part that is accepted by the mill.


(a) Mature cane.

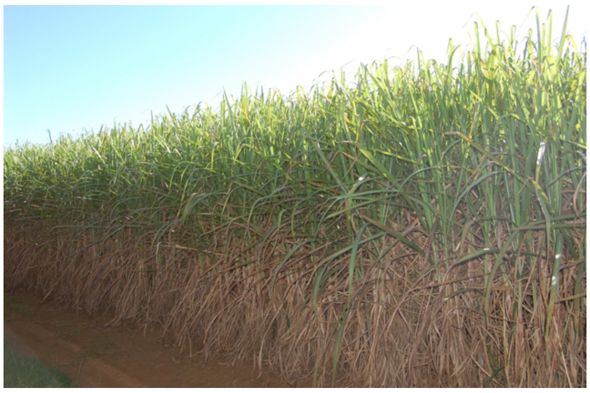

(b) Mature cane.

Figure 3.4: *Cane at ages above twenty months.*

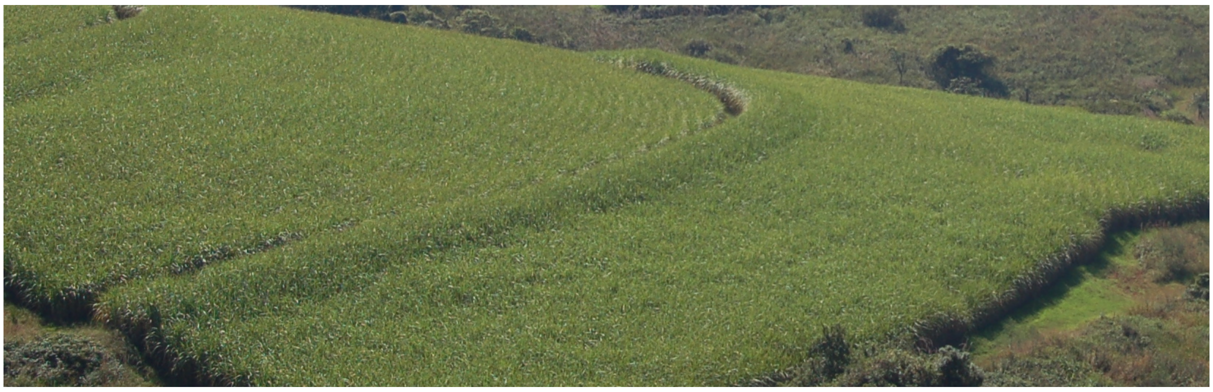


Figure 3.5: *Cane of different ages and varieties.*

the ripener has had time to take effect, and should not be harvested too late [126].

3.4 Sugarcane harvesting

When the harvesting season begins, the successful grower has prepared for the coming nine months by ensuring that fields are planted with different varieties and of different ages, spread out across the farm. The grower must select suitable fields to harvest, and does so with the objective of maximising the profit over the whole season. Depending on how many fields the grower owns, the field selection may be a daily decision process or may occur less frequently. However, all but the very smallest growers must deliver cane every day, which means that the decision process must be completed at least once per week.

The harvesting date of a field is typically not decided upon very far in advance. There are, however, a number of rules-of-thumb that are in use today when making scheduling decisions. Every time that a new field must be harvested, the farm's fields are sorted according to age. Fields that may not be harvested due to physical constraints are postponed, and the oldest of the remaining fields are sorted according to estimated recoverable value. The oldest, most valuable field is selected for harvest, and is burnt some time before cutters or a mechanical harvester arrives. The process of burning is illustrated in Figures 3.7(a) and 3.7(b).

The curved shape of lodged cane causes difficulties during the cutting and loading operation, and there are claims [126] that more soil is inadvertently captured during the loading operation

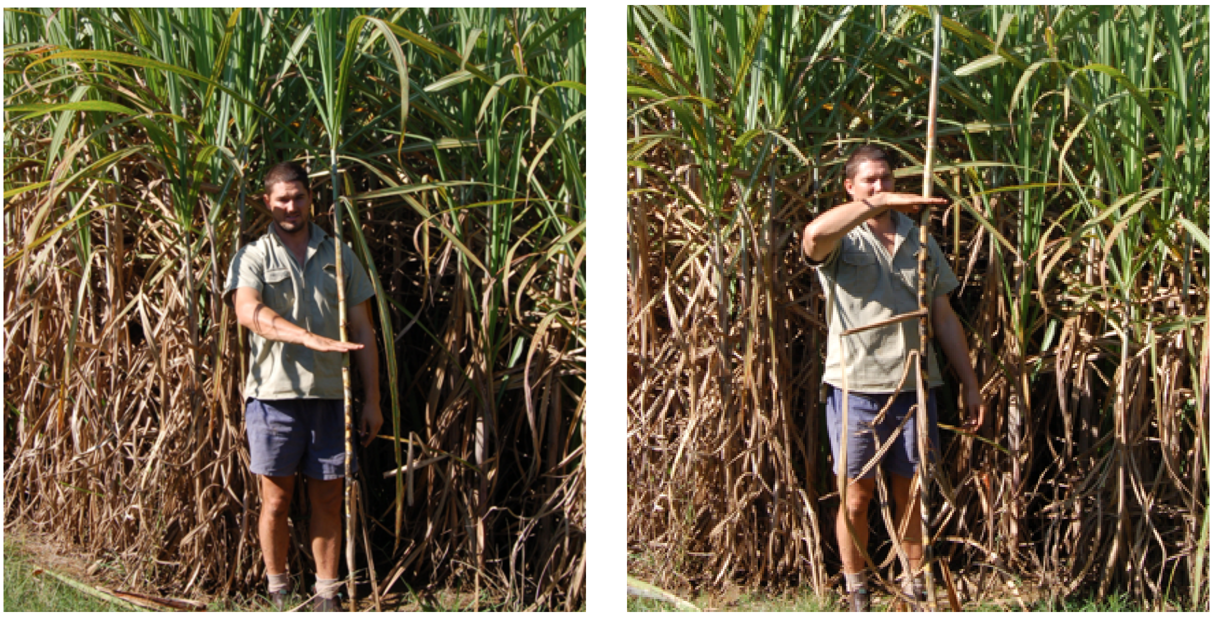

(a) N12. (b) N37.

Figure 3.6: *Cane of different varieties, but of the same age. The N37 variety was more successful than the N12 during the cold and wet growth period of 2007/2008.*

when harvesting a field of lodged cane. The soil causes problems at the mill due to build-up and other interference within the milling process. Furthermore, curved stalks do not stack as neatly as straight cane. One may therefore argue that lodged cane takes longer to load and requires more space on the long-range transport vehicles, thereby reducing transportation efficiency.

It is considered good practice to cut down cane as soon as possible after burning (for example, it is not good to burn a field on Monday and to let it be cut in portions throughout the week. It is better to split the field and burn it in portions as well). Larger farms usually employ larger field sizes, due to their faster cutting rate.

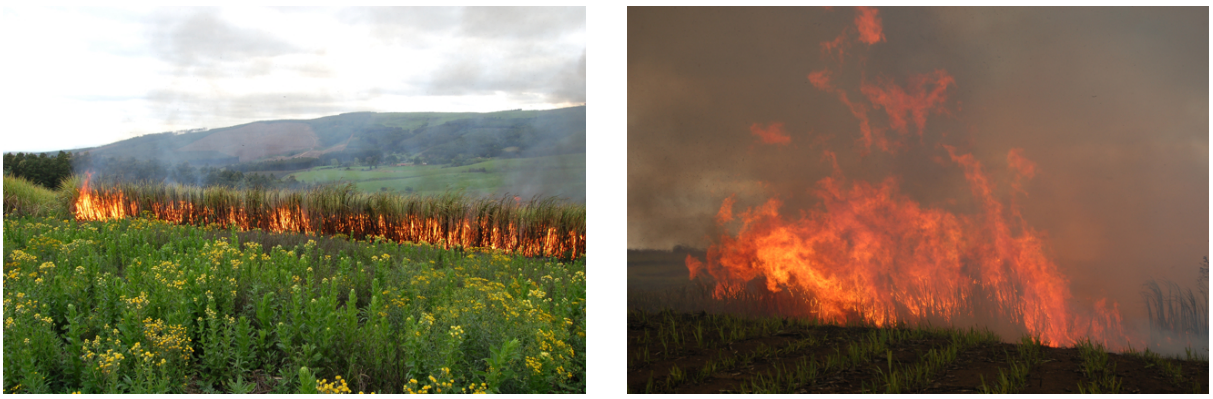
The cutters may enter a field and begin cutting the stalks as soon as the field has cooled down after being burnt. In South Africa, cutters are usually paid a fixed wage plus a bonus amount per tonne that they cut. To be able to record the amount of cane cut by each cutter, they are assigned one row at a time and a foreman records the distance cut. The distance is used to approximate the tonnage cut by each cutter. Figures 3.8(a) and 3.8(b) illustrate the rows system and the cutting technique.

Whilst on the ground, the cane stalks' tops are cut off in order to ensure that these fibrous parts of the stalks are not delivered to the mill, since RV percentage is adversely affected by high fibre content. The "topped" cane may be left in rows on the ground, a practice called *windrowing*, or stacked and bundled using chains. The process of topping is shown in Figures 3.9(a) and 3.9(b).

3.5 Sugarcane loading and transport

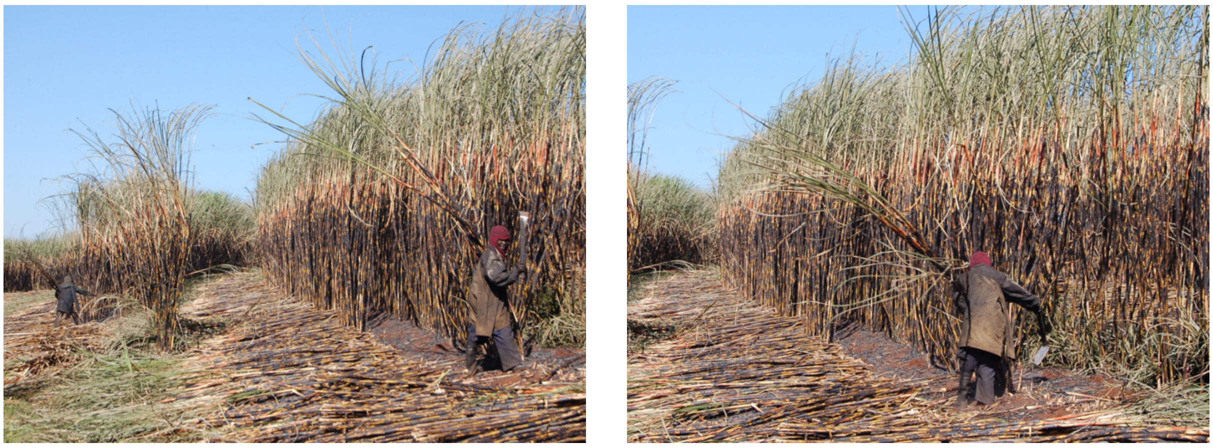
Cane loading is usually mechanised. For example, a machine called a *Bell loader* with a *grab*[3] attached to its lifting arm may be employed to collect the cane and drop it into the hold of a

[3]A *grab* is a term used for a device designed to grab sugarcane.



(a) Burning. (b) Jumping fire.

Figure 3.7: *The process of burning consists of setting fire to a field from its edges and burning it inwards, rather than setting fire to one edge in which case the fire builds up heat while reaching the opposite edge. Wind conditions and adjacency of other fields are other important factors to take into consideration in order to avoid the phenomenon of fire jumping (the spread of fire from one field to an adjacent field.*



(a) Starting position. (b) Striking cane.

Figure 3.8: *Manual cutting of sugarcane. A cutter here holds a handful of stalks which he bends while striking a special kind of machete with a bent tip at the base of the stalks, as close to the ground as possible.*

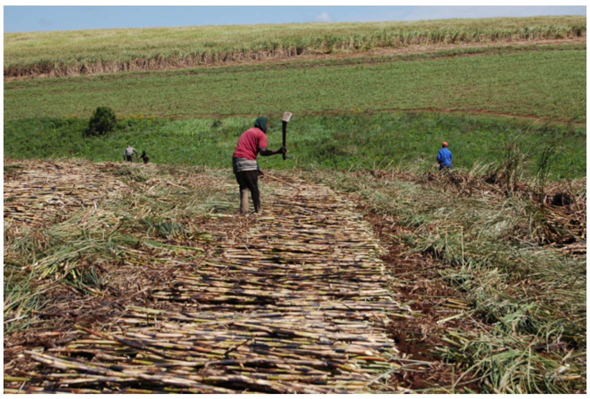
trailer. The trailer may be—in the case of *zone loading*[4]—attached to a tractor or—in the case of *direct haulage*[5]—attached to a long-range transport vehicle. The grab procedure consists of driving the loader forward for a suitable distance with the open grab snugly against the ground, whereupon the loader stops moving and pushes the grabbed cane down and then repeats the process until the grab is full. The grab is then closed and the loader drives to the trailer and unloads the cane. Figures 3.10(a), 3.10(b), 3.12(a) and 3.12(b) illustrate the sequence of events.

Another manner in which to load the harvested cane onto vehicles is by means of so-called bundles. Bundles of cane are held together by steel chains which are lifted by a crane onto transportation vehicles, possibly weighing the load at the same time [179].

Alternatively, a mechanical harvester may be used. The harvester begins by cutting the stalks as

[4]Zone loading is the process of unloading a tractor-trailer onto the ground at a loading zone outside the field and subsequently reloading the cane onto a long-range transport vehicle.

[5]Direct haulage is the process of loading a long-range transport vehicle directly in a field.


(a) Topping of straight cane.

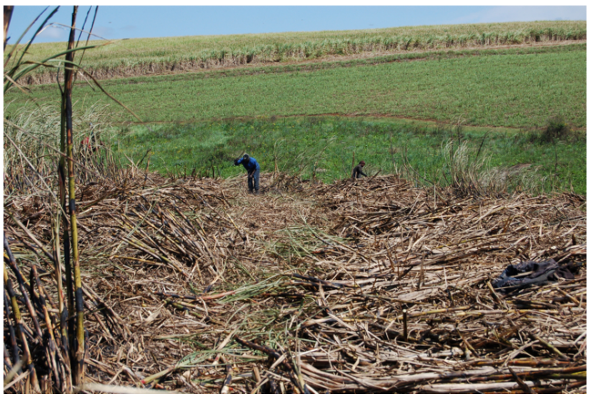
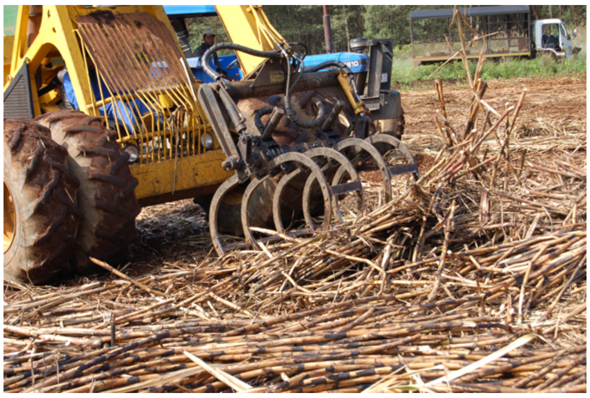

(b) Topping of lodged cane.

Figure 3.9: *Topping of cane is done to remove the topmost part of the stalks since it contains no value in terms of sugar extraction. Topping is clearly more arduous in the case of lodged cane, since the tops then point in different directions, as shown in* (b).


(a) A Bell loader grabs and collects cane from a windrow.

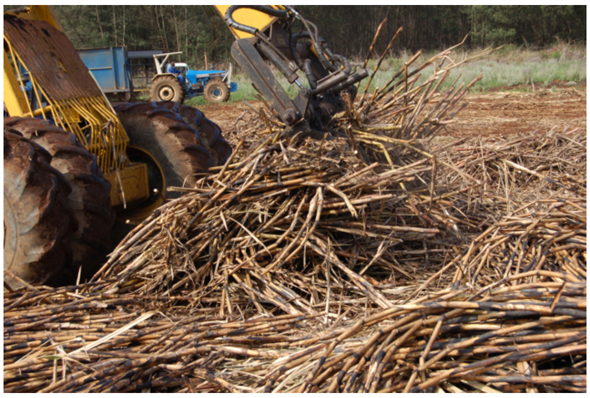

(b) The loader pushes and grips until the grab is full.

Figure 3.10: *The Bell loader is a machine specially designed for grabbing cane.*

close to the ground as possible. Inside the harvester, the stalks are cut into *billets* approximately 30 cm in length and the stalk tops are expelled onto the ground. The billets are offloaded directly from the harvester onto a trailer, usually pulled by a tractor. The use of a mechanical harvester therefore entails zone loading.

The loaded cane is transported to the grower's crush mill, where it is weighed and tested. In Australia, for instance, it is common to transport cane via rail, a practice less common in South Africa [68]. The cane is unloaded onto a *spiller* and that is the point where grower and miller part ways in terms of contractual bonds with regards to physical flow of cane in the supply chain. The spilling operation is shown in Figures 3.13(a) and 3.13(b). The cane continues immediately from the spiller table to a *shredder* where it is turned into a coarse, wet pulp, which constitutes the beginning of the milling process.

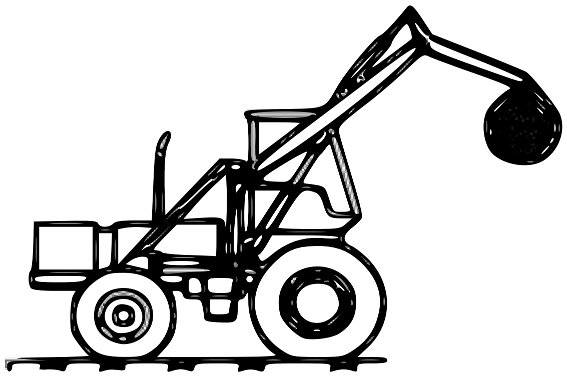


Figure 3.11: *A bundle grab loader (derived from [179]).*

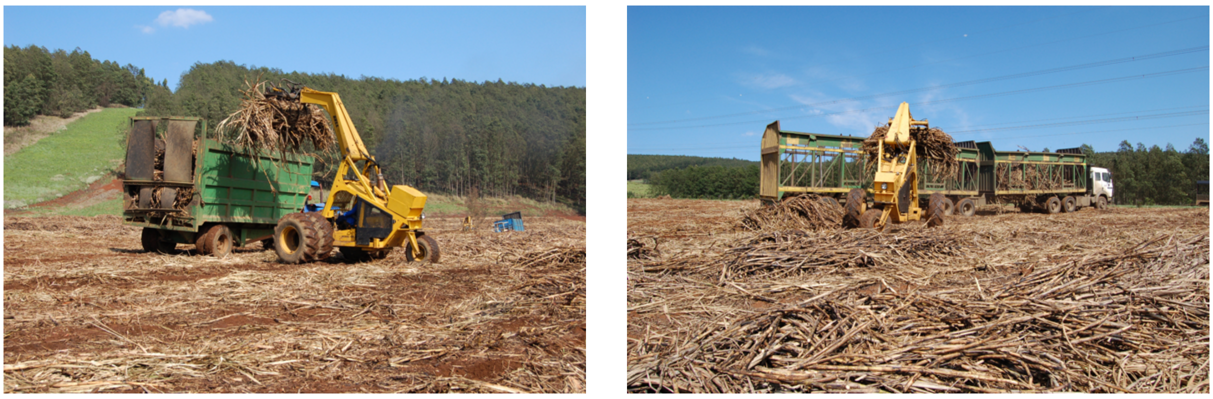

(a) Unloading onto a tractor-trailer combination. (b) Unloading onto a long-range transport vehicle.

Figure 3.12: *A Bell loader is also designed for loading cane onto trailers.*

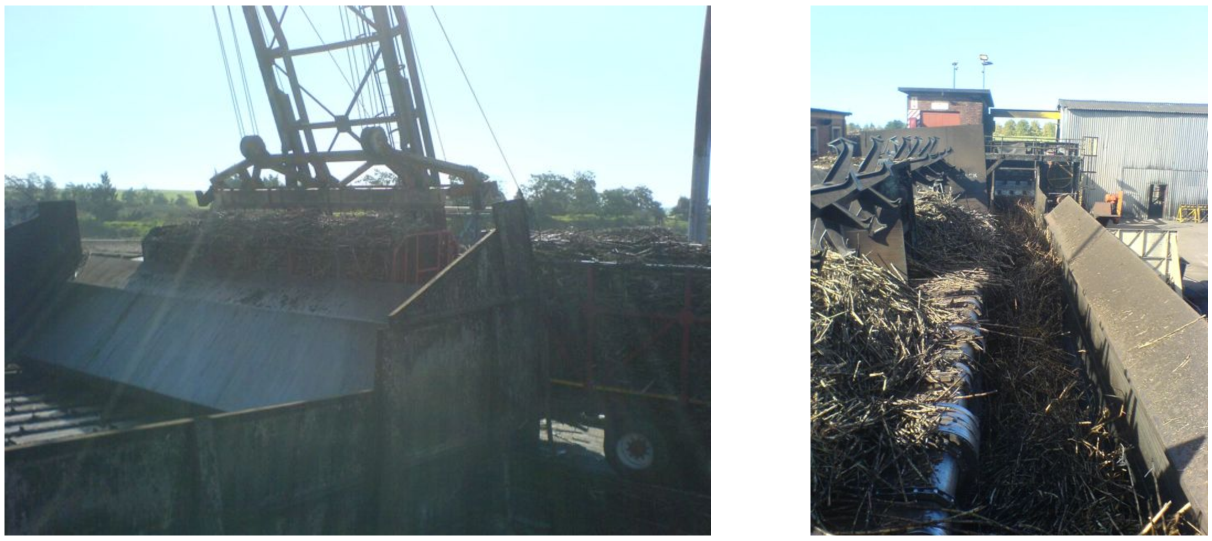

(a) Spilling cane from trailer onto the spiller table at the mill. (b) Spiller table buffering the cane before the shredder.

Figure 3.13: *The spiller operation is the final point of delivery for growers at the mill.*

3.6 Sugarcane constituents and use

According to Almazan *et al.* [2], out of 1 000 kilograms (kg) of mature cane in the field, 824 kg reaches the mill[6] on average, 94 kg is left in the field and 82 kg is waste, cleaned out at an initial cleaning process. In the 824 kg that reaches the milling process, there is, on average, 430 kg of liquid waste, 33 kg of mud, 1 kg of ash, 26 kg of molasses, 231 kg of bagasse and 104 kg sugar. Bagasse is composed of fibre, pith, insoluble solids and water and is used for fuel in the mill, paper production, chip board production and animal feed and several other purposes. Bagasse and sugar may, in turn, be used to generate electricity, and a combination of ethanol, heat energy and electricity through various schemes [3]. The production of ethanol from sugarcane is promising even on a global fuel supply scale, according to Goldemberg *et al.* [78]. Molasses is mainly used for producing fuel ethanol and the white spirit known as *rum* but also in its raw form as cattle feed [2].

3.7 Chapter summary

This chapter contains a brief description of the global centrifugal sugar producers, of which South Africa is the 13th largest in terms of tonnage. Sugarcane cultivation, harvesting, loading and transport activities are described, and important issues are highlighted. The various constituents of a typical sugarcane crop are outlined together with some of their uses. The main result of this chapter is the understanding of how the various processes required to produce sugarcane fit together, and focus is not on a detailed description of each process.

This chapter is presented in partial fulfilment of Dissertation Objectives I, III and IV, and together with Chapter 2 completes the fulfilment of those objectives.

[6]This and the following figures are not clearly referenced by Almazan *et al.* [2], but should however suffice as a basis to judge average proportions between the constituents of cane.

CHAPTER 4

Formal problem description

Contents

The *tactical sugarcane harvest scheduling problem* (THSP) considered in this dissertation is the problem of sugarcane field harvest scheduling in general, and how support may be provided to managers charged with the task of deciding which field to harvest at any given point in time during the harvesting season.

4.1 THSP contexts

There are various possible contexts within which to consider this problem, and those taken into consideration here are *small-scale solitary growers*, *small-scale harvesting groups*, *medium-scale solitary growers*, *medium-scale commercial harvesting groups*, *independent large-scale commercial estates* and *mill-owned large-scale commercial estates.* Henceforth, when referring to the THSP, all of the above contexts are included implicitly.

4.1.1 Small-scale grower contexts

Small-scale growers are here divided into two groups based on whether they pool their DRD with other growers or not. In the former case, the grower has—to a certain extent—surrendered his/her tactical harvesting decision to a contractor who decides which grower's fields to harvest at what time. In the latter case, the grower may choose when to harvest his/her fields according to the prevailing arrangement with the mill. A small-scale grower is here defined by having dedicated less than 10 hectares of land to the production of sugarcane on a permanent basis.

In the sugar industry in South Africa, these growers are often called subsistence growers (see §3.2).

The *small-scale solitary grower* context is characterised by the situation in which the sugarcane grower is independent of other growers and has very few fields to harvest. Harvesting operational capacity is assumed to be managed by the grower, and any decisions related to capacity are thus not considered part of the THSP within this context. The mill is assumed to allow the delivery of cane without adherence to the DRD. The *net realisable operational profit*[1] from harvesting each field is assumed to be a function of time.

The **THSP within the small-scale solitary grower context** *consists of providing decision support to small-scale growers who are charged with scheduling the harvest of a set of sugarcane fields during a single season. The objective is to maximise the profit to the grower while exploiting the possibility of scheduling the fields in a time-wise disjoint sequence.*

The *small-scale harvesting group* context is characterised by the situation where a number of growers have formed a group consisting of small-scale growers. This group of small-scale growers are assumed to have pooled their DRDs and act as a single entity towards the mill. A contractor or haulier often acts as the middleman between such a harvesting group and the mill. The contractor may be a separate business entity or an entity created by the group of growers or may be of some other constitution. The small-scale harvesting group is assumed to agree on a division of the season into several rounds, where each grower's fields are harvested during one or several such rounds. The contractor assumes the responsibility of harvesting and delivering cane to the mill at a consistent rate. When the contractor arrives at some farm, the grower should already have decided which set of fields to harvest. The decision of when to arrive at each farm is, however, not taken by any single decision maker, but rather jointly by the harvesting group.

The **THSP within the small-scale harvesting group context** *consists of providing decision support to a group of small-scale growers charged with scheduling the harvesting of several sets of fields during a single season, where inter-set relations are of major concern. The objective is to maximise the profit for the group while each grower within the group is treated equitably.*

4.1.2 Medium-scale commercial grower contexts

A medium-scale commercial grower is typically characterised by having dedicated between 10 and 1000 hectares of land to the production of sugarcane on a permanent basis. These farms are sometimes referred to as family-owned farms and average approximately 187 hectares devoted to sugarcane production [194]. Medium-scale commercial growers are also divided into two groups based on whether they act singly or in a group setting.

The *medium-scale solitary commercial grower* context is mainly characterised by the proviso that the harvesting operation is managed so that the amount of cane delivered to the mill does not deviate more than 10 % above or below a DRD on a weekly basis. Growers within this context often deliver cane on several fully loaded long-range transport vehicles during each day of the milling-week[2]. Despite harvesting capacity being subject to several conditions, such as labour supply, weather and stochastic availability of equipment (breakdowns, preventive maintenance, *etc.*), the grower is assumed to be able to maintain a constant harvesting rate.

[1]The term *net realisable operational profit* refers to the profit resulting from selling the cane on the field after having deducted harvesting and transportation costs. Henceforth, the term *profit* is used instead of the term net realisable operational profit.

[2]The days of the week during which the mill is crushing.

This is an important assumption in that it is a departure from a typical miller's main short-term objective [47], namely to maintain a steady flow of cane on a *daily basis* to the mill. Assuming that any grower is able to maintain a steady capacity every single day removes the explicit requirement for the DSS to solve problems related to steady delivery volumes of cane. The DSS, however, indirectly addresses this issue since if a manager relies on the scheduling ability of the DSS, then he/she is indirectly required to maintain a steady capacity. One may, however, argue that the incentive is not on a daily level, but rather on a level corresponding to the length of a planning period. There will thus exist cause for concern with respect to the daily delivery consistency, even if the DSS is employed.

The **THSP within the medium-scale solitary commercial grower context** *is the problem of providing decision support to medium-scale commercial growers charged with scheduling the harvesting of a single set of fields which is to be harvested in a continuous fashion during a single harvesting season. The harvesting schedule should maximise the total profit from the harvesting of each field combined, given that the value of each field as well as the cost of harvesting each field vary as functions of time.*

The *medium-scale commercial harvesting group* context is the situation in which there are several medium-scale growers who jointly plan and operate a single or very small number of *harvesting fronts*[3]. Joining such a group enables growers to employ a manager, to share the use of equipment and to co-operate in terms of labour supply problems [25, 40, 126, 139, 214]. Once formed, such a group may manage the harvesting of hundreds of fields. To gain some perspective on the business scale of this context, one may consider that a single field of 10 hectares may contain 1 200 tonnes of cane and that a single unit increase in relative recoverable value percentage in such a field corresponds to approximately R24 000 of profit increase. It is thus safe to make the assumption that no grower within a medium-scale commercial harvesting group accepts a poorer schedule than that which may be achieved solitarily.

The **THSP within the context of medium-scale commercial harvesting groups** *consists of providing decision support to groups of medium-scale commercial growers charged with scheduling the harvesting of a number of sets of fields across a single harvesting season. The potential profit assigned to each field is to be realised to the fullest possible extent, while considering the required adherence to the DRD.*

4.1.3 Large-scale commercial estate contexts

Large-scale commercial estates may be independent of any mill or may be owned by a mill. These two situations constitute the two contexts under which commercial estates are considered.

The *independent large-scale commercial estate* context is characterised by several hundred fields totalling an area of thousands of hectares, owned by a single business entity, but spread over a vast area, even across different mill regions. A large estate presumably employs a manager of the harvesting operation who constructs and executes several harvesting plans, one for each harvesting front. Furthermore, the ownership of a large estate may be divided among several shareholders, separating the harvest scheduling from the owners.

The **THSP within the context of independent large-scale commercial estates** *consists of providing decision support to the manager of the harvesting operation of an estate, who is*

[3]The term *harvesting front* refers to a set of fields that will be harvested one after the other, as well as the equipment and personnel assigned to perform this harvesting, and as a rule, there is no concurrent harvesting of any fields within the same harvesting front.

charged with scheduling a harvesting operation characterised by a large size, a wide array of equipment and personnel in use, potentially long travelling distances, a requirement to adhere to a DRD as well as the division of the harvesting operation into several harvesting fronts.

The *mill-owned large-scale commercial estates* context is characterised by the fact that the two partly opposite inclinations of miller and grower here merge. This interesting situation is different from all of the contexts mentioned above since now the miller may adjust harvesting rates according to the crushing capacity of the mill. In the case of an unreliable mill, this may create a need for relatively frequent updating of harvesting schedules, something that should increase the need for a computerised DSS. Moreover, during major environmental disasters the DSS may provide planning support across an entire disaster area, implementation of the schedules now unimpeded by otherwise necessary lengthy, sometimes fruitless, negotiations. Finally, the harvesting front is, as for the independent large-scale commercial estates, relatively large and travelling costs present an undeniable factor influencing harvest scheduling decisions.

The **THSP within the context of mill-owned large-scale commercial estates** *is the same as for the independent large-scale commercial estates with the exception that mitigation between grower and miller (who are in this context part of the same business entity) is assumed to occur more easily.*

4.2 Dissertation scope revisited

The problem outlined above presents a challenge in that there is a necessity in developing a suitable model under close collaboration with industry professionals. It is foreseen that a further limitation of the scope in §1.4 is required to delimit the otherwise too large group of stake-holders. Therefore the dissertation scope is augmented with the limitation that the DSS will only be designed to model and solve the THSP within the context of *medium-scale commercial harvesting groups.*

It is, however, possible that the DSS will be applicable within other contexts, especially if minor changes are allowed subsequent to development conducted under collaborative terms with the relevant stake-holders. It is not believed that by selecting a particular context, the project will end in an overly specialised DSS computer implementation.

Henceforth, when the THSP is mentioned, context still matters, but when the DSS is mentioned as an answer or solution to the THSP, the context is that of *medium-scale commercial harvesting groups* unless otherwise specified.

4.3 The DSS design and development approach

The proposed approach towards solving the THSP consists of the design and development of a *decision support system framework* as well as an implementation of this framework on a personal computer in a stand-alone fashion which may be run on a Windows-based PC requiring only Excel 2007 or later after having been installed. The DSS is to be based on known predictive modelling and combinatorial optimisation modelling techniques.

4.3.1 The DSS building blocks

The DSS framework should firstly consist of a module of databases for field information, cane yield prediction models, recoverable value percentage prediction models, effects models accounting for extraneous events and coefficients related to harvesting cost prediction models. These databases should be editable by the intended user.

The field database is envisaged to contain a list of each field on the farm or estate accompanied by information on field size, field toposequence[4], field aspect[5], cane variety, crop class, estimated yield, date when last harvested (or date of start of growth in the case of a plant crop), as well as a list of extraneous events which have occurred on each field accompanied by a percentage describing the portion of the crop affected by the extraneous event occurrence. The cane yield prediction models as well as the recoverable value percentage prediction models require their respective parameters to be specified by the user. The parameters may be found by fitting the cane yield or recoverable value percentage prediction models to the available field records' data on cane age, crop class, harvest day and respective cane yield or recoverable value percentage at previous harvests. The extraneous events models as well as the harvesting cost prediction models should have default coefficients incorporated in them, but should be alterable by the user.

The DSS framework secondly should consist of a computational module for predicting the profit from harvesting each field during each of a set of future discrete time periods. This module should not require input from the user but should rather take all its inputs from the databases. The profit values output by this module should be relative estimates, but the cane yield estimate, which is foreseen to be a by-product in this module, is an estimate of the absolute cane yield.

Thirdly, the DSS should contain a scheduling model that schedules the fields into an, according to a specific criterion, optimal harvesting schedule.

Fourthly, the DSS should consist of a user interface module, with user-editable databases, user-editable prediction models, facilities for selecting which fields in the field database should be included in the schedule as well as other schedule-specific information, facilities for running the computational module for predicting the profit from harvesting each field, facilities for running the scheduling algorithm and finally facilities for printing a harvesting schedule as decision support in selecting which field to harvest next.

4.3.2 The DSS validation

The validation of the DSS should occur while it is being revised in response to complaints, suggestions and technical problems, throughout an extended collaborative experiment. The first operational version of the DSS should be ready prior to the onset of the 2009 harvesting season to be put in use within a specific mill area. The data gathering process should be conducted in parallel with readying the DSS, and should also be concluded before the season opening date.

The DSS should be used to generate schedules at regular intervals during the 2009 harvesting season. Hopefully, several harvesting fronts may be shadow scheduled. Development of the DSS is foreseen to occur in parallel with shadow scheduling.

[4]The term *field toposequence* is taken throughout this dissertation to mean the vertical situation of a field, *i.e.* whether the field is located on a hill-top, mid-slope or on a valley bottom.

[5]The term *field aspect* is taken throughout this dissertation to represent the prevailing direction that a field faces.

The results of the validation effort should provide an answer to the question whether tactical sugarcane harvest decision support has been rendered by the DSS implementation. The various components of the DSS are, however, foreseen to require individual validation, before they may be used as valid building blocks in other ventures.

4.4 Chapter summary

In this chapter, the problem to be considered in this dissertation was described in some detail and structured according to various different viewpoints or contexts. These contexts are *small-scale solitary growers*, *small-scale harvesting groups*, *medium-scale commercial solitary growers*, *medium-scale commercial harvesting groups*, *independent large-scale commercial estates* and *mill-owned large-scale commercial estates.*

The planned *decision support system design and development* approach taken towards solving this problem was outlined by listing its major building blocks, requirements imposed on the user and other forms of input. The main components of an envisaged validation process of the DSS were also described.

The thesis of this dissertation is that the THSP in the contexts mentioned in §4.1.1, §4.1.2 and §4.1.3 may be supported by a DSS requiring only readily available data, incorporating well-known predictive modelling and combinatorial optimisation modelling techniques. The planned DSS is foreseen to, at least, provide some evidence towards accepting or discarding this thesis with respect to the medium-scale grower contexts, found in §4.1.2.

Part II

Tactical harvest scheduling decision support system

CHAPTER 5

Optimisation modelling literature review

Contents

A number of well-known problems from the operations research literature, which are all relevant to the THSP, are reviewed in this chapter. As well as highlighting various classical operations research problems related the THSP, this chapter also contains a summary of appropriate exact and approximate solution approaches for these problems.

5.1 Well-known OR problems from the literature

Problems found in the OR literature often share some properties with other problems; the THSP, for example, shares several properties with the *classical assignment problem* and the *travelling salesman problem*. This section contains a description of a set of problems and is aimed at uncovering the similarities between these problems and the THSP.

5.1.1 The classical assignment problem

The *classical assignment problem* (CAP) is concerned with the assignment of a set of agents to a set of tasks such that each agent is assigned to perform exactly one task, each task is assigned to exactly one agent and the total cost of the assignments is minimised. The first time the CAP appeared in the literature was in 1952 in a paper by Votaw and Orden [224]. Other early work on this problem is most notably the development of the Hungarian method by Kuhn as a solution approach for the CAP in 1955 [125]. The literature on problems related to the CAP is large and spans more than 50 years of research [168]. If I^α is the set of agents and J^α is the set of tasks, the CAP may be stated as the problem of

$$\text{minimising} \quad z = \sum_{i \in I^\alpha} \sum_{j \in J^\alpha} c_{ij}^\alpha x_{ij}^\alpha \tag{5.1}$$

subject to the constraints

$$\sum_{i \in I^\alpha} x_{ij}^\alpha = 1, \qquad j \in J^\alpha, \tag{5.2}$$

$$\sum_{j \in J^\alpha} x_{ij}^\alpha = 1, \qquad i \in I^\alpha, \tag{5.3}$$

$$x_{ij}^\alpha \in \{0, 1\}, \qquad i \in I^\alpha, \quad j \in J^\alpha, \tag{5.4}$$

where the decision variable x_{ij}^α takes the value 1 if agent $i \in I^\alpha$ is assigned to perform task $j \in J^\alpha$, or 0 otherwise, and where c_{ij}^α is the cost incurred when agent i is assigned to perform task j. Constraint set (5.2) ensures that each task has exactly one agent assigned to it, whereas set (5.3) ensures that each agent is assigned to perform exactly one task. Finally, constraint set (5.4) ensures the binary nature of the decision variables. The CAP is neither *NP-complete*[1] nor *NP-hard*, but rather belongs to the class of *P* [9, 125, 174].

In the context of the THSP, the tasks of the CAP may be identified with the fields, the agents may be identified with the time periods and the cost coefficients c_{ij}^α may be taken as the negative of the estimated profit from harvesting field i during period j. This formulation is, however, not entirely suitable for the THSP since fields may differ in size, and it may be very difficult to derive a practical harvesting strategy from a solution to the problem if some time periods are only partially utilised or if some fields are only partially harvested during a time period. An iterative approach may be considered, where one first locates an optimal assignment, then adjusts the period lengths in order to fit in the assignments, solves the adjusted problem, readjusts the period lengths and so on, in the hope that the procedure will converge to a solution where each set of assigned fields fits its corresponding harvesting period. This approach is not considered here.

[1] *P* is a class of problems for which there exists algorithms that can find solutions in a polynomial number of steps with respect to problem size. *NP* is a class of problems that ask questions which may be answered with yes or no. For a problem to be part of *NP*, there must exist a method of checking, in a polynomial number of steps with respect to problem size, whether a guessed solution is, indeed, a solution to the problem. An *NP-complete* problem is an *NP* problem for which the existence of an algorithm that is guaranteed to find a solution in a polynomial number of steps is impossible, unless there exists a polynomial algorithm for each problem in *NP* [20]. The Clay Institute offers a USD $1 million prize associated with establishing whether there exists such an algorithm, *i.e.* proving whether $NP = P$ [20, 27].

5.1.2 The generalised assignment problem

A more natural way to model the THSP is to adopt a so-called *generalised assignment problem* (GAP) model formulation. The minimisation form of the GAP is usually also described in terms of a set of jobs and a set of agents. Each job is to be assigned to one of the agents who is to perform the job at some cost while consuming some portion of a limited resource (which may or may not be time). Each agent-job combination is cost-specific as well as resource-specific. Each agent has associated with him/her a specific amount of total available resource, and may thus perform any set of jobs that has a combined resource requirement that is no more than that available to the agent. The GAP appeared for the first time in the literature in the 1975 paper by Ross and Soland [176] as a generalisation of the CAP. In the GAP, agents are, as mentioned earlier, allowed to process multiple jobs, while in the CAP each agent may only process one job. To model agent capacity, a set of resource constraints on the agents is enforced. In Ross and Soland's original formulation, the objective was to

$$\text{minimise} \quad z = \sum_{i \in I^\alpha} \sum_{j \in J^\alpha} c_{ij}^\alpha x_{ij}^\alpha \tag{5.5}$$

subject to the constraints

$$\sum_{j \in J^\alpha} r_{ij} x_{ij}^\alpha \leq b_i, \qquad i \in I^\alpha, \tag{5.6}$$

$$\sum_{i \in I^\alpha} x_{ij}^\alpha = 1, \qquad j \in J^\alpha, \tag{5.7}$$

$$x_{ij}^\alpha \in \{0,1\}, \qquad i \in I^\alpha, \quad j \in J^\alpha, \tag{5.8}$$

where I^α is again the set of agents and J^α is the set of jobs. Here r_{ij} denotes the resource required by agent i to perform job j, while b_i is the resource available for agent i. Furthermore, c_{ij}^α is the cost incurred when agent i is assigned to perform job j and x_{ij}^α is again a binary decision variable taking the value 1 if job i is assigned to agent j, or 0 otherwise. Constraint set (5.6) ensures that the resource available to each agent is not exceeded by the job assignment requirement, while constraint set (5.7) ensures that each job is assigned to exactly one agent. Finally, constraint set (5.8) ensures the binary nature of the decision variables. The GAP is *NP-hard*[2] [178].

The GAP may be used as a model for the THSP by identifying the agents with the time periods and by identifying the tasks with the fields. During each period in the THSP, a maximum amount of sugarcane may be delivered to the mill, represented by the available resource b_i. Each field contains a time-dependent amount r_{ij} of cane to be delivered to the mill when x_{ij} is 1 (*i.e.* when field i is harvested during period j). Since the coefficients c_{ij} are time-dependent, the GAP is able to accommodate the differences in profit from harvesting due to seasonal progression. There are, however, some issues related to practicality arising with this approach. Assuming that a typical harvesting period is two weeks long, of which there may be twenty during the entire harvesting season, the first practical concern when adopting the GAP as a model formulation for the THSP is of a combinatorial nature. Some subset of the fields may be suited for harvest during a particular period from a profit point of view, but their combined total cane yield may be quite different from the total yield required by the contract between the grower and the miller during the period. If, for example, the total combined cane yield is

[2] *NP-hard* is a class of problems in which each problem may be reduced to an *NP-complete* problem in a polynomial number of steps with respect to problem size [20].

500 tonnes and a period requirement is 400 tonnes, the solution to (5.5)–(5.8) is mathematically infeasible since one of the constraints in constraint set (5.6) will be violated by a margin of 100 tonnes. In practice, however, the 100 tonnes may simply be left for harvesting during the following time period, something that cannot be accommodated when using the GAP as a model. This issue impacts negatively on the practical value of a solution to (5.5)–(5.8).

The second practical concern is that of a timeline resolution problem arising if there are a few, relatively large, fields to be scheduled. The time periods must always be made long enough to accommodate the largest field (otherwise (5.5)–(5.8) will have no feasible solution). The value of cane on the fields changes too rapidly to be approximated at intervals longer than one month. For example, if the time periods are one month long and two fields are scheduled to be harvested during some period, there is at least a two week scheduling precision deficit concerning the particular point in time at which the harvesting of each field should commence, since the order in which the two fields should be harvested is not considered by the GAP. In terms of scheduling fields that suffer from adverse extraneous events, this scheduling precision deficit may be too large.

5.1.3 The asymmetric travelling salesman problem

Another approach towards modelling the THSP is to consider arranging the set of fields into a sequence representing the order in which sugarcane fields are to be harvested, in which case the celebrated *travelling salesman problem* (TSP) is the natural starting point for model development. The TSP is traditionally concerned with the order in which a salesman should visit a set of pre-specified cities, in order to minimise travel cost. Each city should be visited exactly once in such a *tour* and a cost is associated with travelling between each pair of cities. The TSP is sometimes called the *symmetric* travelling salesman problem, due to the travelling cost between two cities being independent of the direction of travel. The first integer programming formulation of the TSP is due to Dantzig *et al.* [45] in 1954.

The generalisation of the TSP in which the travelling costs depend on the direction of travel is called the *asymmetric travelling salesman problem* (ATSP). The ATSP is described here, highlighting the TSP as a special case of the ATSP. Consider a set of vertices V and a set of weighted arcs A which form a complete, directed, simple, weighted graph $G(V, A)$. Each pair of vertices $u, v \in V$ is joined by two arcs, one for each direction of travel between the vertices. The arcs are denoted by $uv \in A$ and $vu \in A$, respectively. Travelling along an arc $uv \in A$ is associated with a weight, denoted by c^{β}_{uv}. The arc weights satisfy the triangle inequality, which requires a direct connection between any two vertices to be of less than or equal weight than any indirect connection between the two vertices via a third vertex. A feasible tour visits each vertex in $G(V, A)$ exactly once, returning to the vertex from which the tour started. An optimal tour is a feasible tour with minimum total weight. The objective of the ATSP is therefore to

$$\text{minimise} \qquad z = \sum_{u \in V} \sum_{v \in V} c^{\beta}_{uv} x^{\beta}_{uv} \tag{5.9}$$

subject to the constraints

$$\sum_{v\in V} x^{\beta}_{uv} = 1, \qquad u \in V, \tag{5.10}$$

$$\sum_{u\in V} x^{\beta}_{uv} = 1, \qquad v \in V, \tag{5.11}$$

$$\sum_{u\in S^{\beta}} \sum_{v\in S^{\beta}} x^{\beta}_{uv} \leq \left|S^{\beta}\right| - 1, \qquad S^{\beta} \subset V, \quad \left|S^{\beta}\right| \geq 2 \tag{5.12}$$

$$x^{\beta}_{uv} \in \{0,1\}, \qquad u, v \in V, \tag{5.13}$$

where the proper subset S^{β} represents a so-called *sub-tour* (a closed route that does not visit all vertices) through the graph $G(V, A)$, and the binary decision variable x^{β}_{uv} takes the value 1 if the arc uv is traversed, or 0 otherwise. Constraint set (5.10) ensures that the tour leaves each vertex exactly once, while constraint set (5.11) ensures that the tour arrives at each vertex exactly once. Furthermore, constraint set (5.12) ensures that all sub-tours are forbidden, while constraint set (5.13) maintains the binary requirement on the decision variables. To avoid loops, the convention is adopted of setting $c^{\beta}_{uu} = \infty$ for all $u \in V$. The TSP is equivalent to (5.9)–(5.13) if $c^{\beta}_{uv} = c^{\beta}_{vu}$ for all vertex pairs $u, v \in V$.

The cities, or vertices, in the ATSP may be identified with the fields in the THSP. The ATSP would have been an interesting option for modelling the THSP if the main concern had been between which fields travelling occurs, but it is more important to record at what *time* the travelling occurs, since that indicates the time of harvest. In the context of travelling salesman problems, time is by convention taken to be equivalent to cost. The THSP requires these two dimensions to be distinct, so that harvesting costs and revenue may be modelled as functions of time.

5.1.4 The vehicle routing problem

According to Laporte [130], the classical *vehicle routing problem* (VRP) was introduced by Dantzig and Ramser in 1959 [46] and is concerned with constructing m routes of minimum total weight for m vehicles through a weighted, undirected graph $G(V, E^{\beta})$ with vertex set $V = \{0, 1, 2, \ldots, n\}$ and weighted edge set E^{β}. Each vehicle has a capacity Q, and each customer, modelled by a vertex $u \in V \setminus \{0\}$, is associated with a nonnegative demand q^{β}_{u} which may not exceed Q. Each vehicle may only complete one route, starting out and returning to a central depot modelled by the vertex $0 \in V$, and each customer must be visited exactly once by some vehicle. An optimal solution to the problem consists of m such routes through $G(V, E^{\beta})$, with a minimum combined weight. The VRP is a generalisation of the TSP and if $Q = \infty$ and $m = 1$, they are equivalent. The following formulation of the VRP is that of Laporte [130].

Let the integer decision variable x^{β}_{uv} denote the number of times the edge $uv \in E^{\beta}$ is traversed in a feasible solution to the VRP. The variable x^{β}_{0v} may take the value 0, 1 or 2, but for any other vertex u, x^{β}_{uv} may take only the values 0 or 1. This construct allows for a return trip between vertex 0 and any vertex $v \in V \setminus \{0\}$, which occurs if a route only visits a single customer. The objective in the VRP is to

$$\text{minimise} \qquad z = \sum_{uv\in E^{\beta}} c^{\beta}_{uv} x^{\beta}_{uv} \tag{5.14}$$

subject to the constraints

$$\sum_{v\in V\setminus\{0\}} x^{\beta}_{0v} = 2m, \tag{5.15}$$

$$\sum_{u<w} x^{\beta}_{uw} + \sum_{v>w} x^{\beta}_{wv} = 2, \qquad w \in V\setminus\{0\}, \tag{5.16}$$

$$\sum_{u\in S^{\beta}, v\notin S^{\beta}} x^{\beta}_{uv} \geq \left\lceil \sum_{u\in S^{\beta}} q^{\beta}_{u}/Q \right\rceil, \qquad S^{\beta}\subset V\setminus\{0\}, \tag{5.17}$$

$$x^{\beta}_{uv} \in \{0,1\}, \qquad u,v\in V\setminus\{0\}, \tag{5.18}$$

$$x^{\beta}_{0v} \in \{0,1,2\}, \qquad v\in V\setminus\{0\}, \tag{5.19}$$

where c^{β}_{uv} is the cost of traversing the edge $uv \in E^{\beta}$ and S^{β} is any proper subset of the customer set $V\setminus\{0\}$. Constraint set (5.15) ensures that there are two edges incident to the depot vertex 0 per route. Constraint set (5.16) ensures that two edges are incident to each vertex other than the depot, one being the "incoming" edge and the other the "outgoing" edge. Constraint set (5.17) prevents sub-tours by forcing all proper subsets of the set of customer vertices to be connected to the depot, as well as ensuring that the vehicle capacities are not violated. The term $\lceil \sum_{u\in S^{\beta}} q^{\beta}_{u}/Q\rceil$ in (5.17) is a lower bound on the number of vehicles required to service all customers in S^{β}. Constraint sets (5.18)–(5.19) ensure integrality and place bounds on the decision variables.

In the context of the THSP, the VRP may be applicable if there are several harvesting fronts to consider where some maximum cutting capacity is associated with each harvesting front. In such a case, the harvesting fronts would correspond to the vehicles, the cutting capacity to the vehicle capacity and the fields to the customers. The weakness of this modelling approach is, again, the time-dependency of the field harvesting profits in the THSP, which is not accommodated in the VRP. However, it should be mentioned that a vehicle routing problem with time windows appears in the literature (see, for example, Soler *et al.* [190]), which would allow for a time-dependent multiple-harvesting front THSP model formulation.

5.1.5 The time-dependent travelling salesman problem

The earliest formulation of the *time-dependent travelling salesman problem* (TDTSP) was published in 1960 in a paper by Miller *et al.* [153]. According to Fox *et al.* [59] its origin may, however, be traced to a paper by Bowman [21] in 1956. The problem is concerned with constructing an optimal tour through a directed, weighted graph $G(V,A)$ of order n with vertex set V, arc set A and with arc costs depending on arc positions in the tour. Define the binary decision variable $x^{\gamma}_{uv\alpha}$ to take the value 1 if the arc $uv \in A$ is traversed as the α^{th} arc in the tour, or 0 otherwise. Then the TDTSP is the problem of

$$\text{minimising} \qquad z = \sum_{u\in V}\sum_{v\in V}\sum_{\alpha\in V} c^{\gamma}_{uv\alpha} x^{\gamma}_{uv\alpha} \tag{5.20}$$

subject to the constraints

$$\sum_{v\in V}\sum_{\alpha\in V} x^{\gamma}_{uv\alpha} = 1, \qquad u \in V, \tag{5.21}$$

$$\sum_{u\in V}\sum_{\alpha\in V} x^{\gamma}_{uv\alpha} = 1, \qquad v \in V, \tag{5.22}$$

$$\sum_{u\in V}\sum_{v\in V} x^{\gamma}_{uv\alpha} = 1, \qquad \alpha \in V, \tag{5.23}$$

$$\sum_{v\in V}\sum_{\alpha\in V\setminus\{1\}} \alpha x^{\gamma}_{uv\alpha} - \sum_{v\in V}\sum_{\alpha\in V} \alpha x^{\gamma}_{vu\alpha} = 1, \qquad u \in V\setminus\{1\} \tag{5.24}$$

$$x^{\gamma}_{uv\alpha} \in \{0,1\}, \qquad u,v,\alpha \in V, \tag{5.25}$$

where it is assumed that each traversal of an arc requires one time period and that the tour begins at vertex 1 during period 1 and returns to vertex 1 during period n. Constraint set (5.21) ensures that exactly one arc enters each vertex in such a tour, while constraint set (5.22) ensures that exactly one arc exits each vertex. Furthermore, constraint set (5.23) ensures that exactly one arc is traversed during each period, while constraint set (5.24) forbids the formation of sub-tours and advances time by one period per traversed arc. The final constraint set (5.25) is a binary constraint on the decision variables.

If sugarcane farms generally consisted of equally sized fields, the TDTSP would have been a suitable model formulation in the context of the THSP, where the fields would then be represented by the vertices of the graph, the negative of the profits of harvesting each field during each period would be represented by the cost coefficients in (5.20), and the time periods of the THSP would be the periods of the TDTSP. One might further investigate the implications of assuming that all fields are of an equal, average size; this special case of the THSP is, however, not considered in this dissertation.

5.1.6 The ATSP with time windows

The *asymmetric travelling salesman problem with time windows* (ATSPTW) is an ATSP with the additional requirement that each vertex must be visited within a certain time window.

Consider a set of vertices V and a set of weighted arcs A which form a complete, directed, simple, weighted graph $G(V, A)$. Each pair of vertices $u, v \in V$ is joined by two arcs, one for each direction of travel between the pair of vertices. The arcs are again denoted by $uv \in A$ and $vu \in A$, respectively. Travelling along an arc $uv \in A$ is associated with a weight, denoted by c^{β}_{uv}. A time window $[a^{\beta}_{u}, b^{\beta}_{u}]$ is associated with each vertex $u \in V$. The tour must leave each vertex within its associated time window. The tour may arrive at vertex u before time a^{β}_{u} but will in such a case leave the vertex at time a^{β}_{u}. Traversing the arc $uv \in A$ implies a travel time t^{β}_{uv} and a travel cost c^{β}_{uv}, regardless of the time at which the traversal commences. The ATSPTW is the problem of

$$\text{minimising} \qquad z = \sum_{u\in V}\sum_{v\in V} c^{\beta}_{uv} x^{\beta}_{uv} \tag{5.26}$$

subject to the constraints

$$\sum_{v \in V} x^{\beta}_{uv} = 1, \qquad u \in V, \tag{5.27}$$

$$\sum_{u \in V} x^{\beta}_{uv} = 1, \qquad v \in V, \tag{5.28}$$

$$y^{\beta}_{u} + t^{\beta}_{uv} - M\left(1 - x^{\beta}_{uv}\right) \leq y^{\beta}_{v}, \qquad u, v \in V, \tag{5.29}$$

$$a^{\beta}_{u} \leq y^{\beta}_{u} \leq b^{\beta}_{u}, \qquad u \in V, \tag{5.30}$$

$$x^{\beta}_{uv} \in \{0, 1\}, \qquad u, v \in V, \tag{5.31}$$

where the decision variable x^{β}_{uv} takes the value 1 if the tour progresses from vertex u to vertex v and the decision variable y^{β}_{u} denotes the time at which the tour leaves vertex u. Constraint set (5.27) ensures that the tour exits each vertex exactly once, constraint set (5.28) ensures that the tour enters each vertex exactly once. In (5.29), M is a number larger than b^{β}_{u} and the constraint sets (5.29) and (5.30) together ensure that the time windows are adhered to. Finally, constraint set (5.31) ensures that the decision variable x^{β}_{uv} is binary.

In terms of the THSP, the ATSPTW is applicable in cases where one considers assigning each field a period of the season during which it *must* be harvested. This may be the case, for example, when the optimal time to harvest a field is relatively certain, while the size of the harvesting front or distance between the fields is large. Such a situation may, however, place more importance on the travelling costs than is necessary. The THSP assumes that the harvesting front returns to a depot every night, which renders the harvesting sequence irrelevant with respect to travelling costs.

5.1.7 The ATSP with time-dependent costs

The *asymmetric travelling salesman problem with time-dependent costs* (ATSPTDC) is a variation of the ATSP in which inter-city travelling costs depend on how much time has elapsed since the start of the tour, *i.e.* the sequence of vertices preceding the traversal of a particular arc determines arc-cost. According to Albiach *et al.* [5], the ATSPTDC is an extension of the ATSPTW and it may be formulated as follows. Let $V = \{0, 1, 2, \ldots, n, n+1\}$ be the vertex set and let A be the arc set of a graph $G(V, A)$ in which there is one arc for each time instant and each direction between each pair of vertices. The vertices 0 and $n+1$ are a dummy starting depot vertex and a dummy ending depot vertex, respectively, for any feasible tour. A time window $[a^{\delta}_{u}, b^{\delta}_{u}]$ is associated with each vertex $u \in V$, where a^{δ}_{u} is the earliest time at which vertex u may be visited in a feasible solution tour and b^{δ}_{u} is the latest time by which the vertex may be visited. These windows are defined in terms of time instants $\tau^{\kappa}_{u} = a^{\delta}_{u} + \kappa - 1$, $\kappa \in \left\{1, 2, \ldots, b^{\delta}_{u} - a^{\delta}_{u} + 1\right\}$. The parameters $c^{\delta}_{uv\kappa}$ and $t^{\delta}_{uv\kappa}$ denote the cost and time, respectively, of travelling to vertex v from vertex u, starting at time instant κ. Define the decision variable $x^{\delta}_{uv\kappa}$ to take the value 1 if the arc uv is traversed starting at time instant κ, or 0 otherwise. Furthermore, let $p^{\delta}_{uv\kappa}$ denote a waiting penalty for arriving early at vertex v from vertex u, having exited u at time instant κ, let the decision variable y^{δ}_{u} denote the time instant at which the tour leaves vertex u and let $w^{\delta}_{uv\kappa}$ denote the number of time instants by which the tour arrives early at vertex v. Finally, let $\mathbb{Z}^{+}$ denote the set of all positive integers. Then the objective in the ATSPTDC is to

$$\text{minimise} \qquad z = \sum_{u \in V} \sum_{v \in V} \sum_{\kappa \in \left\{1, 2, \ldots, b^{\delta}_{0}\right\}} \left(c^{\delta}_{uv\kappa} x^{\delta}_{uv\kappa} + p^{\delta}_{uv\kappa} w^{\delta}_{uv\kappa}\right) \tag{5.32}$$

subject to the constraints

$$\sum_{v\in V}\sum_{\kappa\in\left\{1,2,\ldots,b_0^\delta\right\}} x_{uv\kappa}^\delta = 1, \qquad u\in V\setminus\{n+1\}, \tag{5.33}$$

$$\sum_{u\in V}\sum_{\kappa\in\left\{1,2,\ldots,b_0^\delta\right\}} x_{uv\kappa}^\delta = 1, \qquad v\in V\setminus\{0\}, \tag{5.34}$$

$$\sum_{v\in V}\sum_{\kappa\in\left\{1,2,\ldots,b_0^\delta\right\}} x_{n+1,v,\kappa}^\delta = 0, \tag{5.35}$$

$$\sum_{u\in V}\sum_{\kappa\in\left\{1,2,\ldots,b_0^\delta\right\}} x_{u,0,\kappa}^\delta = 0, \tag{5.36}$$

$$x_{uv\kappa}^\delta = 0, \qquad u=v\in V, \quad \kappa\in\left\{1,2,\ldots,b_0^\delta\right\}, \tag{5.37}$$

$$x_{0,n+1,\kappa}^\delta = 0, \qquad \kappa\in\left\{1,2,\ldots,b_0^\delta\right\}, \tag{5.38}$$

$$y_0^\delta = 1, \tag{5.39}$$

$$y_{n+1}^\delta - 1 = b_0^\delta, \tag{5.40}$$

$$y_u^\delta + t_{uv\kappa}^\delta - M\left(1 - x_{uv\kappa}^\delta\right) \leq y_v^\delta, \qquad u,v\in V, \quad \kappa\in\left\{1,2,\ldots,b_0^\delta\right\}, \tag{5.41}$$

$$\sum_{v\in V}\sum_{\kappa\in\left\{1,2,\ldots,b_0^\delta\right\}} \kappa x_{uv\kappa}^\delta = y_u^\delta, \qquad u\in V\setminus\{n+1\}, \tag{5.42}$$

$$a_u^\delta \leq y_u^\delta \leq b_u^\delta, \qquad u\in V, \tag{5.43}$$

$$a_v^\delta - y_u^\delta - t_{uv\kappa}^\delta - M\left(1 - x_{uv\kappa}^\delta\right) \leq w_{uv\kappa}^\delta, \qquad u,v\in V, \quad \kappa\in\left\{1,2,\ldots,b_0^\delta\right\}, \tag{5.44}$$

$$x_{uv\kappa}^\delta \in \{0,1\}, \qquad u,v\in V, \quad \kappa\in\left\{1,2,\ldots,b_0^\delta\right\}, \tag{5.45}$$

$$y_u^\delta \in \mathbb{Z}^+, \qquad u\in V, \tag{5.46}$$

where M is a number larger than b_u^δ, and the objective function (5.32) consists of a sum of the travelling costs and the waiting penalty costs. Constraint set (5.33) ensures that each vertex is exited exactly once in a solution tour, while constraint set (5.34) ensures that each vertex is entered exactly once. Constraint sets (5.35) and (5.36) ensure that the dummy ending depot vertex is not exited and that the dummy starting depot vertex is not entered, (5.37) forbids looping within each vertex, while (5.38) ensures that the dummy ending depot vertex is not visited immediately after the dummy starting depot vertex. Constraint sets (5.39)–(5.42) ensure that the time instant when the tour leaves each vertex is computed correctly and that the tour does not arrive at a vertex after the last time instant of its associated time window, while constraint set (5.43) ensures that the tour does not leave any vertex at a time instant outside of its associated time window. Constraint set (5.44) computes the waiting time $w_{uv\kappa}^\delta$ (expressed as a number of time instants) spent at vertex v, having left directly from vertex u at time instant κ. Finally, constraint set (5.45) ensures that the decision variable $x_{uv\kappa}^\delta$ is binary and constraint set (5.46) ensures that the decision variable y_u^δ is a positive integer.

The ATSPTDC is capable of modelling time-dependency both in terms of travel time and travel cost. This makes the ATSPTDC very attractive in the context of the THSP, since the yield and RV % of a sugarcane crop change with time. If the ATSPTDC is used to model the THSP, $c_{uv\kappa}^\delta$ may be taken as the negative of the profit from harvesting field v at time instant κ directly after having completed the harvesting of field u (which occurs at time instant $\kappa - t_{uv\kappa'}^\delta$, where κ' is the time instant at which the harvesting operation began harvesting the previous field).

The time required to harvest field v at time instant κ directly after having finished harvesting field u may be denoted by $t^{\delta}_{uv\kappa}$. Both $c^{\delta}_{0v\kappa}$ and $t^{\delta}_{0v\kappa}$ are zero for $\kappa \in \left\{1, 2, \ldots, b^{\delta}_{0}\right\}$ and the time required to harvest a field includes the time required to travel to the next field. Despite its modelling strengths, the ATSPTDC is not used in its entirety in this dissertation, since time windows are not considered necessary for modelling the THSP and since travel times in the THSP need not depend on time. It was, however, the main inspiration for one of the modelling approaches towards solving the THSP adopted later in this dissertation.

5.1.8 The job-shop scheduling problem

The *classical job-shop scheduling problem* (CJSP) commonly arises in the manufacturing environment and was introduced by Bowman in 1959 [22]. Consider a set of jobs where each job is to be processed on a set of machines and where each machine can process one job at a time. Each job j must be processed on each machine i in a certain order, and the objective is to minimise the total makespan (the total time elapsed between the start of the first job and the completion of the last job). The set N^{ϵ} contains all operations (i, j) (indicating that job j is to be processed on machine j) and the set A^{ϵ} contains all precedence constraints (h, i, j) indicating that job j must be processed on machine i before being processed on machine h. The following formulation of the CJSP is due to Pinedo [171, pp. 85–86]. The objective is to

$$\text{minimise} \qquad z = C_{\max} \tag{5.47}$$

subject to the constraints

$$\begin{aligned}
y^{\epsilon}_{hj} - y^{\epsilon}_{ij} &\geq p^{\epsilon}_{ij}, & & (h, i, j) \in A^{\epsilon}, & & (5.48)\\
C_{\max} - y^{\epsilon}_{ij} &\geq p^{\epsilon}_{ij}, & & (i, j) \in N^{\epsilon}, & & (5.49)\\
y^{\epsilon}_{ij} - y^{\epsilon}_{ik} \geq p^{\epsilon}_{ik} \quad &\text{or} \quad y^{\epsilon}_{ik} - y^{\epsilon}_{ij} \geq p^{\epsilon}_{ij}, & & (i, k), (i, j) \in N^{\epsilon}, \quad i = 1, 2, \ldots, m, & & (5.50)\\
y^{\epsilon}_{ij} &\geq 0, & & (i, j) \in N^{\epsilon}, & & (5.51)
\end{aligned}$$

where the decision variable y^{ϵ}_{ij} denotes the time at which job j is processed on machine i and p^{ϵ}_{ij} denotes the processing time duration required for job j on machine i. Constraint set (5.48) ensures that each job is processed on the machines according to the precedence requirements, while constraint set (5.49) computes the makespan $C_{\max}$. The constraints in (5.50) are called *disjunctive constraints*[3] and ensure that jobs which are to be processed on the same machine are ordered in some fashion so as to disallow simultaneous processing (the leftmost disjunctive constraint provides for job j to precede job k and the rightmost disjunctive constraint provides for job k to precede job j, where either ordering is acceptable). Finally, constraint set (5.51) ensures the non-negativity of the decision variable.

If a harvesting front is modelled as an operation consisting of various parts (such as burning, cutting, loading, zone unloading, zone loading and transportation), then a set of fields considered for harvesting during the season may be identified with the jobs and the various harvesting front parts may be identified with the machines of the CJSP. In such a case the minimisation of the makespan would correspond to attempting to maximise the utilisation of the available resources (by squeezing the harvesting of fields into as short a time frame as possible). Adopting an iterative approach whereby the available resources are manipulated until a solution is reached for which the makespan is equal to the length of the milling season, a theoretically optimal

[3]Disjunctive constraints function in pairs where either one must be satisfied, whereas conjunctive constraints must all be satisfied.

burning capacity, cutting capacity, loading and transportation capacity may thus be computed. This approach may be useful for large operations controlled by a single business entity.

5.1.9 The sequence and time-dependent scheduling problem

A *sequence and time-dependent scheduling problem* (STDSP), which is relevant to the THSP, was put forward in 2008 by Stecco *et al.* [206, 207], who considered different formulations of the STDSP as well as both exact and metaheuristic solution approaches. This particular scheduling problem is similar to a version of the TDTSP, described above as the ATSPTW. The STDSP is interesting, firstly because it represents one of the most complicated makespan problems within the research area of single machine scheduling and, secondly, because it provides for an alternative modelling approach in the context of the THSP.

The STDSP is the problem of minimising the total makespan that occurs during the processing of a set $N^\zeta = \{0, 1, 2, \dots, n+1\}$ of jobs, where job j has a processing time of p_j^ζ $(j \in N^\zeta \backslash \{0, n+1\})$ and where two different setup times for a particular job depend on which job preceded it as well as on the completion time of the preceding job. Here, 0 and $n+1$ are two dummy jobs with zero processing times; these jobs represent the virtual start and end of the processing schedule. The entire schedule is designed for a single planning horizon which is composed of several planning periods of equal length. The horizon begins at time τ^ζ. For notational purposes, $P^\zeta = \{0, 1, 2, \dots, n\}$ and $S^\zeta = \{1, 2, \dots, n+1\}$ denote the sets of possible predecessors and successors, respectively. Between the processing of any two jobs, two different setups may take place: an *unrestricted* setup may occur at any time, except when the *restricted* setup has been begun and halted (due to insufficient time to complete it), while a restricted setup may only take place outside a forbidden interval $[a^\zeta, b^\zeta] \subset [0, d^\zeta]$ where d^ζ is the end of each planning period. A practical setting in which the restricted setup occurs is in complicated operational environments where skilled people or technology may not be available twenty-four hours a day, so that job changes which require these resources are restricted to parts of the day when the resources are available. Stecco *et al.* [206] made the assumption that the forbidden interval may take the form $[a^\zeta, d^\zeta]$, without loss of generality. The restricted setup time r_{ij}^ζ $(i, j \in N^\zeta)$ between two jobs combined with the unrestricted setup time u_{ij}^ζ $(i, j \in N^\zeta)$ introduces a time-dependent aspect to the problem, since there is an implicit waiting time involved whenever the completion of the restricted setup must wait until the end of a forbidden interval.

An instance of the STDSP is specified by presenting a weighted, directed graph G, whose vertices represent the jobs, while the traversal of an arc represents the transition from one job to the next in the schedule. The total time to traverse an arc is $r_{ij}^\zeta + u_{ij}^\zeta$ if the traversal begins within either the interval $[0, a^\zeta - r_{ij}^\zeta]$ or the interval $[d^\zeta - u_{ij}^\zeta, d^\zeta]$ of any planning period, while the total traversal time is $r_{ij}^\zeta + u_{ij}^\zeta + d^\zeta - a^\zeta$ if the traversal begins within the interval $(a^\zeta - r_{ij}^\zeta, d^\zeta - u_{ij}^\zeta)$. These three intervals constitute the set of relevant intervals K^ζ. The graph G is specified in such a manner that each pair of vertices is joined by six arcs: one arc in each direction for each $k \in K^\zeta$. This arc set is denoted by A^ζ inducing the graph $G = G(N^\zeta, A^\zeta)$. The decision variable t_{ij}^ζ is the departure time when leaving vertex $j \in N^\zeta$ if it was visited immediately after vertex $i \in N^\zeta$. The decision variable h_{ij}^ζ is the number of the planning period during which vertex $j \in N^\zeta$ is to be visited if visited immediately after vertex $i \in N^\zeta$, while the binary decision variable x_{ijk}^ζ takes the value 1 if vertex $i \in N^\zeta$ is exited during relevant interval $k \in K^\zeta$ towards vertex $j \in N^\zeta$, or 0 otherwise. The time required to traverse the arc ijk is denoted by c_{ijk}^ζ and an upper bound on the relevant time interval $k \in K^\zeta$ for arc ijk is denoted by I_{ijk}^ζ. The departure time from vertex 0 is denoted by t_0^ζ. The formulation of the STDSP given here is the final of the three formulations in [206, pp. 2637–2642][4] and consists of

$$\text{minimising} \qquad z = \sum_{i=1}^{n} t_{i,n+1}^\zeta \tag{5.52}$$

subject to the constraints

$$\sum_{i\in P^\zeta}\sum_{k\in K^\zeta} x_{ijk}^\zeta = 1, \qquad j \in S^\zeta, \tag{5.53}$$

$$\sum_{j\in S^\zeta}\sum_{k\in K^\zeta} x_{ijk}^\zeta = 1, \qquad i \in P^\zeta, \tag{5.54}$$

$$\sum_{j\in S^\zeta} t_{ij}^\zeta \geq \sum_{\ell\in P^\zeta} t_{\ell i}^\zeta + \sum_{j\in S^\zeta}\sum_{k\in K^\zeta}\left(c_{ijk}^\zeta + p_j^\zeta\right)x_{ijk}^\zeta, \qquad i = 1,2,\ldots,n, \tag{5.55}$$

$$\sum_{j=1}^{n} t_{0j}^\zeta \geq t_0^\zeta + \sum_{j=1}^{n}\sum_{k\in K^\zeta}\left(c_{0jk}^\zeta + p_j^\zeta\right)x_{0jk}^\zeta, \tag{5.56}$$

$$\sum_{\ell\in P^\zeta} t_{\ell i}^\zeta \geq \sum_{j\in S^\zeta}\sum_{k\in K} I_{ij,k-1}^\zeta x_{ijk}^\zeta + d^\zeta \sum_{m\in P^\zeta} h_{mi}^\zeta, \qquad i = 1,2,\ldots,n, \tag{5.57}$$

$$t_0^\zeta \geq \sum_{j=1}^{n}\sum_{k\in K^\zeta} I_{0j,k-1}^\zeta x_{0j,k-1}^\zeta, \tag{5.58}$$

$$\sum_{\ell\in S^\zeta} t_{\ell i}^\zeta \leq \sum_{j\in S^\zeta}\sum_{k\in K^\zeta} I_{ijk}^\zeta x_{ijk}^\zeta + d^\zeta \sum_{m\in S^\zeta} h_{mi}^\zeta, \qquad i = 1,2,\ldots,n, \tag{5.59}$$

$$t_0^\zeta \leq \sum_{j=1}^{n}\sum_{k\in K^\zeta} I_{0jk}^\zeta x_{0jk}^\zeta, \tag{5.60}$$

$$t_{ij}^\zeta \leq M_1^\zeta \sum_{k\in K^\zeta} x_{ijk}^\zeta, \qquad i \in P^\zeta, \quad j \in S^\zeta, \tag{5.61}$$

$$h_{ij}^\zeta \leq M_2^\zeta \sum_{k\in K^\zeta} x_{ijk}^\zeta, \qquad i \in P^\zeta, \quad j \in S^\zeta, \tag{5.62}$$

$$t_0^\zeta \geq \tau^\zeta, \tag{5.63}$$

$$h_{ij}^\zeta \in \mathbb{Z}^+, \qquad i \in P^\zeta, \quad j \in S^\zeta, \tag{5.64}$$

$$x_{ijk}^\zeta \in \{0,1\}, \qquad i \in P^\zeta, \quad j \in S^\zeta, \quad k \in K^\zeta. \tag{5.65}$$

Constraint set (5.53) ensures that the tour enters each vertex (except vertex 0) exactly once, while constraint set (5.54) ensures that the tour departs from each vertex (except vertex $n+1$) exactly once. Constraint set (5.55) computes the time at which the tour departs from each vertex (except the vertex from which the tour departed after having visited vertex 0), while constraint (5.56) computes the time at which the tour departs from the vertex that it entered after departing from vertex 0. Constraint sets (5.57) and (5.59) force the correct relevant interval to be used for the decision variable x_{ijk}^ζ, based on the time at which the tour leaves each vertex, except for the relevant intervals applicable when leaving vertex 0, which is accounted for by constraints (5.58) and (5.60). The variables x_{ijk}^ζ, t_{ij}^ζ and h_{ij}^ζ are linked in constraint sets (5.61) and (5.62), where M_1^ζ may be given the value of any upper bound on $\sum_{i=1}^{n} t_{i,n+1}^\zeta$ and M_2^ζ may be

[4]The authors presented a non-linear programming formulation, a linear TDTSP formulation and a stronger, integer programming formulation (the one appearing here).

given the value of any upper bound on $\sum_{i=1}^{n} t^{\zeta}_{i,n+1}/d^{\zeta}$. The final three constraint sets, namely (5.63), (5.64) and (5.65), ensure that the tour starts within the planning horizon $[\tau^{\zeta}, \infty]$, that the planning periods are real, positive integers, and that x^{ζ}_{ijk} is a binary integer, respectively.

In the context of the THSP, the vertices in an instance of the STDSP may represent the fields while each of the arcs from one vertex to the next may represent the various relevant periods of a typical harvesting day (recall that there are k arcs emanating from each vertex to the next, except from vertex $n+1$). The harvesting of a field requires burning, which may only be performed during the light hours of the day (due to fire-safety considerations). Manual cutting also requires light, and the least costly option is the sun, while it is also preferable to complete cutting during the morning. Loading of cane and transportation of equipment can occur throughout the day and so these operations may be classified as unrestricted setup times. In this context, a field is said to be harvested by the series of setups: burning (restricted to daylight), cutting (restricted to the morning), loading (unrestricted) and transport (unrestricted). The main realism problem with this model in the context of the THSP is that an actual harvesting operation may be burning one field while another is being cut or loading at one field while another is being cut. In fact, all the operations above may overlap, which disqualifies the formulation for the THSP as is. As a provision for situations where the operations may not overlap, the formulation should, however, not be discounted. As a final note, the above example illustrates the difficulty with which optimisation may be applied to the operational problems of sugarcane harvesting. The STDSP as well as all other scheduling formulations encountered in the literature review of this dissertation lack the important ability to account for cost-time dependency in the sense required by the THSP. The realm of scheduling is generally not concerned with costs other than the cost of time. For examples of sequence-dependent scheduling problems (in which the sequencing of jobs affects the processing time of each job), see [1, 60, 61, 124, 127, 129, 212, 225].

5.2 Popular exact solution methodologies

Instances of the problems reviewed in §5.1 have been solved exactly by researchers using various methods. One of the most common exact[5] solution approaches is the *branch-and-bound* method, often incorporating the generation of *cutting planes*. These methods are described in this section.

5.2.1 The branch-and-bound method

In this section, the general framework of the branch-and-bound method is first described, after which more specific implementations of the method for integer programming problems and the TSP in particular are considered.

The general approach

The branch-and-bound method is a general procedural framework within which a solution to a combinatorial optimisation problem, denoted by $\mathcal{C}$, may be proven to be optimal, given that it is possible to relax $\mathcal{C}$ into a solvable problem $\mathcal{C}'$ and subsequently separate $\mathcal{C}'$ into subproblems, denoted by $\mathcal{S}'_i$, where the index i identifies the subproblems. Furthermore, the combined solution space of all subproblems must cover that of the original problem and it must be possible to

[5]The term *exact* in this context means that the solution method is guaranteed to uncover the optimal solution to the problem, given enough time and computational resources (often theoretical amounts of resources).

place upper and lower bounds on the objective function values of solutions to $\mathcal{S}_i'$ for all i. The separation into subproblems is termed *branching* since the data structure most often used to record the progress of the method resembles a tree, called the *branch-and-bound tree* or the *search tree*. If $\mathcal{C}$ is a minimisation problem, achieving a low objective function value during the search is better than achieving a high objective function value. If a subproblem's lower bound on the optimal objective function value is larger than the smallest upper bound in the search-tree (the current upper bound), the subproblem cannot possibly be used to improve on the current best upper bound, in which case it is *pruned* (discarded) during subsequent stages of the method; this process is known as *bounding*. If all subproblems have been pruned, or if further branching into subproblems is impossible, or if the lower bound of the search tree equals the current upper bound, the current upper bound is the optimum objective function value of the original combinatorial optimisation problem $\mathcal{C}$ and the corresponding solution is proven to be optimal.

The branch-and-bound method for integer programming

In the case of general integer programming problems, the branch-and-bound method is the overwhelmingly most common exact solution approach adopted in the literature. The branch-and-bound method described above in general is easily adapted to the case of a general linear integer programming setting. Consider an integer programming problem with the objective of

$$\text{minimising} \qquad \boldsymbol{c}^T\boldsymbol{x} \tag{5.66}$$

subject to the constraints

$$\boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b}, \tag{5.67}$$

$$\boldsymbol{x} \in \mathbb{Z}_+^n, \tag{5.68}$$

where $\boldsymbol{A}$ is an $m \times n$ matrix of coefficients whose rows correspond to m constraints and whose columns correspond to n integral decision variables captured in an n-column vector $\boldsymbol{x} = [x_1, \ldots, x_n]^{\mathrm{T}}$. In (5.66), $\boldsymbol{c} = [c_1, \ldots, c_n]^{\mathrm{T}}$ is an n-column vector typically containing cost coefficients. The n-column vector of right-hand-side constants $\boldsymbol{b}$ contains lower bound values for each constraint.

The so-called linear programming relaxation problem associated with (5.66)–(5.68) is the problem of

$$\text{minimising} \qquad \boldsymbol{c}^T\boldsymbol{x} \tag{5.69}$$

subject to the slightly weaker constraints

$$\boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b}, \tag{5.70}$$

$$\boldsymbol{x} \geq \boldsymbol{0}, \tag{5.71}$$

where the only difference between the original problem (5.66)–(5.68) and the relaxation (5.69)–(5.71) is the replacement of (5.68) by (5.71).

The branch-and-bound method for the problem (5.66)–(5.68) begins by solving (5.69)–(5.71), then selecting a solution component x_i with a fractional solution value ξ to branch on, subsequently solving the linear programming relaxations of the two subproblems (5.69)–(5.71) together with the additional constraint $x_i \leq \lfloor \xi \rfloor$, and (5.69)–(5.71) together with the additional constraint $x_i \geq \lceil \xi \rceil$. This process is repeated and at any point a solution with no fractional components is a solution to the problem (5.66)–(5.68). The best such solution uncovered at any point in time during an application of the method is called the incumbent, its objective function value constituting an upper bound on (5.66)–(5.68). If a solution is found that has no fractional components and its objective function value is less than the upper bound, it is taken as the incumbent and its objective function value is taken as the upper bound. If a subproblem has an objective function value greater than the upper bound, it is pruned.

A lower bound on the objective function value of the original problem (5.66)–(5.68) is found by taking the lowest objective function value among subproblems that have not yet been branched on. The procedure is terminated whenever the lower bound becomes equal to the upper bound or comes within a predetermined percentage from the upper bound. The linear programming relaxations are sometimes strengthened by the addition of cutting planes, based on certain properties of the matrix $\boldsymbol{A}$, a procedure which is described in §5.2.2.

The branch-and-bound method for the TSP

The TSP presented as problem (5.9)–(5.13) with the addition of the condition $c^{\beta}_{uv} = c^{\beta}_{vu}$ (the condition that changes the ATSP into the TSP) may be relaxed into a CAP by removing the subtour constraint set (5.12). The CAP may be solved efficiently by means of existing algorithms [233], such as the Hungarian Method presented by Kuhn [125] in 1955.

This description of the branch-and-bound method for the TSP is based on that of Winston [233]. The root problem (the original TSP problem), denoted by $\mathcal{C}$, of the branching tree is first relaxed into a root problem relaxation (a CAP), denoted by $\mathcal{C}'$, which is subsequently solved using some available efficient algorithm, such as the Hungarian Method (which renders an integer solution to the CAP). The objective function value of the optimal solution to $\mathcal{C}'$ is taken as the lower bound on $\mathcal{C}$. The solution to $\mathcal{C}'$ is then inspected for subtours and if it does not contain any subtours, *i.e.* does not violate any constraint in the set (5.12), the solution to $\mathcal{C}'$ is also an optimal solution to $\mathcal{C}$, and the method terminates.

If there exists subtours in the solution to $\mathcal{C}'$, one of those subtours is selected to constitute the basis for branching. Within this branching subtour, a number of decision variables are currently 1, indicating the presence of their corresponding arcs in the branching subtour. By choosing one of these variables and forcing it to take the value 0, a subproblem[6], denoted by $\mathcal{S}'_1$, results in which the particular branching subtour cannot possibly exist. By choosing a different variable in the branching subtour and forcing this variable to take the value 0, a different subproblem $\mathcal{S}'_2$ results. For each variable in the branching subtour, one possible subproblem thus exists, and the branching subtour cannot possibly appear in any of them. Each branching step in the method may thus be completed by generating one subproblem for each arc present in the selected branching subtour.

An appropriate subproblem $\mathcal{S}'_i$ to be solved is selected based on some rationale, such as a *depth-first strategy*[7]. The selected subproblem $\mathcal{S}'_i$ is first solved and, as for $\mathcal{C}'$, if the objective function value of $\mathcal{S}'_i$ is larger than the upper bound on that of $\mathcal{C}$ (if no incumbent exists the upper bound is taken to be infinite), $\mathcal{S}'_i$ is pruned. If the objective function value is smaller than the upper

[6]A subproblem to $\mathcal{C}$ inherits the relaxed status from $\mathcal{C}'$ during the branching step in this method, as do the next level of subproblems from the subproblems to $\mathcal{C}'$, and so on.

[7]The root problem $\mathcal{C}$ being the "top" problem, branching downwards in the tree rather than sideways, logically leads to the rapid strengthening of the relaxations, increasing the probability of uncovering a solution without subtours [233].

bound, and if there are no subtours in the subproblem, the objective function value is taken as the new upper bound on $\mathcal{C}$ and no further branching takes place on $\mathcal{S}'_i$. If the objective function value is smaller than the upper bound and there are subtours in the solution to $\mathcal{S}'_i$, branching may take place (in the same manner as for $\mathcal{C}'$).

In the event of a new upper bound being established as a result of uncovering a feasible solution to the root problem, the entire branching tree is inspected, and each subproblem that has an objective function value larger than the new upper bound is pruned.

The current lower bound on $\mathcal{C}$ is the smallest objective function value among the subproblems that have neither been branched on completely nor been pruned. The method terminates if all subproblems are pruned or if the lower bound meets the upper bound.

Some strategic considerations of the method may be highlighted by this brief description of an instance of an application of the branch-and-bound method for the TSP described above. Since the subproblems $\mathcal{S}'_i$ inherit the relaxed status of their "parent" subproblems with the ultimate "grandparent" being $\mathcal{C}'$, the relaxation of the original problem $\mathcal{C}$, any subproblem may be solved to optimality. Suppose that there are several subproblems in the branching tree which have not been solved and several which have been solved. Suppose further that a particular subproblem $\mathcal{S}'_i$ is selected at some point and solved. If a new incumbent is uncovered when solving $\mathcal{S}'_i$, some subproblems in the branching tree may be pruned, including $\mathcal{S}'_i$. If no new incumbent was uncovered, but the solution to $\mathcal{S}'_i$ is devoid of subtours, $\mathcal{S}'_i$ may be pruned, but no other subproblems may be pruned. If there are subtours in the solution to $\mathcal{S}'_i$, and the objective function value is smaller than the upper bound, three main approaches may be adopted: either $\mathcal{S}'_i$ is subjected to branching immediately, generating new subproblems, or one of the other solved subproblems in the branching tree is selected and branched on, or one of the unsolved subproblems is selected and solved as was just described.

5.2.2 The generation of cutting planes

Balas *et al.* [8] attribute the method of generating cutting planes to Gomory [71]. The method was not regarded by the research community as being very useful until the early 1990s, but is now part of some of the solver engines in popular off-the-shelf mathematical programming software such as IBM's ILOG CPLEX Optimizer [107], LINDO System's LINGO [140] and GAMS Development Corporation's GAMS [63].

The general approach

Consider a combinatorial optimisation problem $\mathcal{C}$, and suppose that $\mathcal{C}$ may be relaxed into an easily solved relaxation $\mathcal{C}'$. Initially, an optimal solution to $\mathcal{C}'$ is usually not a feasible solution to $\mathcal{C}$ and is usually not easily transformable into one. The method of cutting planes is based on the idea of strengthening a relaxation so that its optimal solution becomes more likely to be feasible to the original problem. In general, the strengthening consists of the addition of so-called *valid inequalities* to $\mathcal{C}'$, which are constraints obviously satisfied by any feasible solution to $\mathcal{C}$ and hence their addition to $\mathcal{C}'$ excludes some part of the solution space of $\mathcal{C}'$ that does not contain any feasible solutions to $\mathcal{C}$. These superfluous parts of the feasible domain of $\mathcal{C}'$ may be cut off by repeatedly adding valid inequalities (also called cuts), hopefully eventually leading to an improved relaxation whose optimal solution is feasible (and hence optimal) to $\mathcal{C}$.

The generation of cutting planes for the TSP

Exactly as in §5.2.1 the TSP is presented as problem (5.9)–(5.13) with the addition of the condition $c^{\beta}_{uv} = c^{\beta}_{vu}$, and is relaxed into the CAP by removing constraint set (5.12). In addition, this relaxation is further relaxed by substituting constraint set (5.13) with the linear bounds

$$0 \leq x^{\beta}_{uv} \leq 1, \qquad u, v \in V, \tag{5.72}$$

which renders a linear programming relaxation of the CAP, constituted by (5.9)–(5.11), (5.72) and $c^{\beta}_{uv} = c^{\beta}_{vu}$. Let the TSP and the linear programming relaxation of the CAP be denoted by $\mathcal{C}$ and $\mathcal{C}'$, respectively.

The generation of cutting planes for the TSP is described as follows. First the problem $\mathcal{C}'$ is solved. If an uncovered optimal solution to $\mathcal{C}'$ violates some constraint in (5.12), that single constraint is added to $\mathcal{C}'$ which is subsequently resolved. This is repeated until a stopping criterion is satisfied, which may be a time limit or a situation in which the solution to $\mathcal{C}'$ does not violate any constraint in (5.12). If the optimal solution to $\mathcal{C}'$ does not violate any constraint in (5.12), the objective function value is the best lower bound on $\mathcal{C}$ obtainable using this procedure. If the solution is also an integer solution, it is an optimal solution to $\mathcal{C}$. The latter is often not the case, and so the question arises of how to continue the search for a good (or optimal) solution to $\mathcal{C}$, once the cutting plane procedure has been terminated. One answer to this question is to combine the cutting plane procedure with a branch-and-bound procedure, discussed previously.

The branch-and-cut method for the TSP

The *branch-and-cut* method for the TSP is a combination of the branch-and-bound method and the method of generating of cutting planes. The following description is focussed on the main traits of one of several versions of the branch-and-cut method, and these main traits are from Cook *et al.* [39].

Suppose that the cutting plane generation method described above is applied to the TSP, which is again denoted by $\mathcal{C}$. During iteration $i = 1$, $\mathcal{C}$ is relaxed into a linear programming relaxation of the CAP, denoted by $\mathcal{C}'$, by replacing the binary constraints (5.13) with the linear bounds (5.72) as well as removing the subtour elimination constraints (5.12).

In a cutting plane generation phase, cuts are generated for $\mathcal{C}'$ until no subtour elimination constraints are violated. If the solution to $\mathcal{C}'$ at this stage is integer, it is an optimal solution to $\mathcal{C}$. If the solution to $\mathcal{C}'$ is not integer, one of the variables x^{β}_{uv} with fractional values is selected to branch on. Two subproblems result, denoted by $\mathcal{S}'_i$ and $\mathcal{S}'_{i+1}$ respectively, setting $x^{\beta}_{uv} = 0$ in the first and $x^{\beta}_{uv} = 1$ in the second. The cuts that were generated for $\mathcal{C}'$ are carried over to $\mathcal{S}'_i$ and $\mathcal{S}'_{i+1}$.

One of the subproblems, denoted by $\mathcal{S}'_s$, is selected and solved to optimality using a linear programming solution method, such as the simplex method, invented by George Dantzig [44] in 1947, and if the objective function value is larger than the upper bound, $\mathcal{S}'_s$ is pruned, as described for the branch-and-bound method. If the objective function value is smaller than the upper bound, the cutting plane generation phase is completed for $\mathcal{S}'_s$ in the same fashion as for $\mathcal{C}'$. If, during this phase, the objective function value becomes larger than the upper bound, $\mathcal{S}'_s$ is pruned. If the cutting plane generation phase ends with an integer solution to $\mathcal{S}'_s$, this solution is taken as the new incumbent. If the cutting plane generation phase ends with a fractional solution, branching is performed on $\mathcal{S}'_s$ in the same fashion as for $\mathcal{C}'$.

The branch-and-cut method therefore consists of selecting a subproblem, solving the subproblem, generating new cuts for the subproblem, pruning the subproblem if appropriate, updating lower bounds as subproblems are enumerated and solved in the branching tree, updating the upper bound when integer solutions are found that contain no subtours and iterating through this loop until terminating the method when the upper bound is equal to the lower bound.

Good selections of which subproblem to branch on and which variable to fix are crucial to the practical success of any branch-and-bound or branch-and-cut implementation. One subproblem-picking strategy is to pick the subproblem with the minimum objective function value among the subproblems of the search tree, which aims at increasing the lower bound at every subproblem. Another subproblem-picking strategy is to follow a depth-first strategy, which aims at finding an integer solution which may or may not be a tour, the goal being to improve the upper bound (if it is, indeed, a tour). The problem of picking a variable (or arc) to fix may be based on the subproblem solution values for all arcs (except the ones which have already been fixed), where that arc is picked which has a value closest to 0.5, the rationale being that this often causes the greatest difference in optimal objective function value between the current subproblem and the two resulting subproblems. The traversal cost of an arc plays a similar role since fixing an expensive arc generally causes a larger change in objective function value than fixing an inexpensive arc.

5.3 Approximate solution methodologies

Some of the most popular approximate[8] solution methodologies for hard combinatorial optimisation problems (such as those outlined in §5.1) include *ant colony optimisation algorithms* [32, 61, 64, 216], *genetic algorithms* [37, 115, 219, 237], *memetic algorithms* [35, 60], *scatter search algorithms* [12, 105], *simulated annealing algorithms* [41, 167, 173] and *tabu search algorithms* [13, 123, 169] as well as highly problem-specific methods such as advanced versions of the Lin-Kernighan heuristic for the TSP [91, 92] and special bounding techniques such as the *minimum spanning tree bound* for the TSP due to Held and Karp [90]. These methods are not explored in great detail here; however, the method of *tabu search* is given some special attention due to its incorporation into the DSS presented later in this dissertation. The other methods are touched upon briefly towards the end of this section.

5.3.1 The tabu search

Tabu search as a solution approach is relatively simple to implement and has worked well for several instances of the TSP and its generalisations as well as for integer programming problems in general. Laporte [130] provides a review on solution methodologies for the VRP in which he mentions the high success rate of tabu search[9]. In 1990, Laguna *et al.* [127] investigated several tabu search approaches towards solving the single machine scheduling problem. They could solve problems with 25 jobs to optimality, and found solutions to problems with 25–35 jobs that had objective function values equal to that of the best known solutions. More recently, a tabu search was successfully applied by Stecco *et al.* in 2009 [207] in the context of the STDSP. The solution times for problems with 50 jobs ranged from 17 to 145 seconds, while these solutions

[8]The term *approximate* in this context means that the solution method is not designed to prove the exact optimality of the objective function value of the best solution encountered.

[9]Laporte calls a tabu search with attributes an *attribute-based tabu search.*

where between 0.000 % and 2.298 % from a lower bound[10] with the tabu search having been coded in C and run on a 3.0 GHz computer with 1 GB RAM.

The general approach

The method of tabu search was invented during the late nineteen-seventies by Fred Glover [72] who first described it in detail in 1989 [73, 74]. The notation and terminology used in the description presented here for the general methodology is based on that of Glover [73, 74].

As in the descriptions of the exact solution methods in §5.2.1 and §5.2.2, let $\mathcal{C}$ denote a hard combinatorial optimisation problem. Let Ω denote the entire feasible domain of $\mathcal{C}$ and let $\boldsymbol{\omega}$ denote an element of Ω. The core of the tabu search consists of repeatedly performing a so-called *move*, here denoted by s. Performing a move implies that a solution $\boldsymbol{\omega}$ is altered to form another solution, and this changed solution is denoted by $s(\boldsymbol{\omega})$. There may be several types of moves, which differ by the manner in which they alter $\boldsymbol{\omega}$; here s belongs to a single (arbitrary) such *move type*, denoted by S. The set of possible moves depends on which solution $\boldsymbol{\omega}$ is under consideration, and this set is denoted by $S(\boldsymbol{\omega})$. The set $S(\boldsymbol{\omega})$ is also described as the neighbourhood of $\boldsymbol{\omega}$ induced by the move type S. Let the objective function value of $\boldsymbol{\omega}$ be denoted by $V(\boldsymbol{\omega})$.

The tabu search is an iterative method, and is typically divided into three phases, which are described here under the assumption that the search has completed a certain number of iterations of the loop constituted by these three phases and is considered to be in iteration h.

The tabu search always maintains a single *current solution*, denoted by the previously found solution $\boldsymbol{\omega}$, and a *best solution* found so far, denoted by $\boldsymbol{\omega}^\star$. The central phase towards improving the current solution $\boldsymbol{\omega}$ may be called *the neighbourhood enumeration phase* and involves the generation of a set of *trial solutions*, a trial solution being denoted by $\boldsymbol{\omega}'$. In this phase, some or all of the solutions obtainable by applying the move type S on the current solution $\boldsymbol{\omega}$ (*i.e.* some or all of the solutions $\boldsymbol{\omega}' \in \{s(\boldsymbol{\omega}) : s \in S(\boldsymbol{\omega})\}$) are generated.

The next phase may be called the *trial solution selection phase* and here the move s associated with the best trial solution generated during the neighbourhood enumeration phase is compared to *the reversals of previously performed moves*, denoted by s_i^{-1} (where $i < h$ is an identifier of the i^{th} time this phase was applied), which are stored in a so-called *tabu list*, denoted by T. If s appears as one of the inverted moves s_i^{-1} in T, the move s is said to be *tabu* and may not be performed unless $V(\boldsymbol{\omega}')$ is better than $V(\boldsymbol{\omega}^\star)$. If the move was declared to be *tabu*, the next best $\boldsymbol{\omega}'$ is inspected in the same fashion, repeatedly, until a solution $\boldsymbol{\omega}'$ is found that does not have its associated move s in T. This solution is denoted by $\boldsymbol{\omega}'_h$ and the associated move is denoted by s_h.

In the next phase, called the *updating phase*, the finally approved trial solution $\boldsymbol{\omega}'_h$—within the trial solution selection phase—is taken by the tabu search as the new current solution $\boldsymbol{\omega}$ (*i.e.* $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega}'_h$). The move s_h used to generate this solution is reversed into move reversal s_h^{-1} which is subsequently entered together with the identifier $i = h$ into the tabu list T. If T has reached its maximum length, its so-called *tenure*, the oldest reversal of a move is deleted from the list.

This three phase iteration process continues until some stopping criterion is satisfied, such as a time limit or a maximum number of iterations has been reached or, perhaps preferably, until

[10]The lower bound was computed using a linear programming relaxation with the addition of cutting planes, as explained in [206].

$V(\boldsymbol{\omega}^\star)$ is within some percentage of a bound on the best possible objective function value. This tabu search method is illustrated in pseudo-code form in Algorithm 5.1.

Algorithm 5.1: Tabu search algorithm.

Input: A starting solution $\boldsymbol{\omega}$, a neighbourhood, denoted by $S(\boldsymbol{\omega})$, an objective function $V(\boldsymbol{\omega})$ defined for a problem $\mathcal{C}$, a bound on the best objective function value which must be achieved before termination, denoted by *bound*, and a tabu list tenure, denoted by *tenure*.
Output: The best solution $\boldsymbol{\omega}^\star$ found to $\mathcal{C}$.

```
$\boldsymbol{\omega}^\star \leftarrow \boldsymbol{\omega}$;
$h \leftarrow 1$;
$T \leftarrow \emptyset$;
while $V(\boldsymbol{\omega}^\star)$ is not better than bound do
    $trial\_solution\_set \leftarrow \{s(\boldsymbol{\omega}) : s \in S(\boldsymbol{\omega})\}$;
    $trial \leftarrow tabu$;
    while $trial = tabu$ do
        $\boldsymbol{\omega}' \leftarrow s(\boldsymbol{\omega}) \in trial\_solution\_set$ that has the best $V(s(\boldsymbol{\omega}))$;
        if $s \in T$ then
            $trial\_solution\_set \leftarrow trial\_solution\_set \setminus \{s(\boldsymbol{\omega})\}$;
        else
            $trial \leftarrow not\_tabu$;
            $\boldsymbol{\omega}'_h \leftarrow \boldsymbol{\omega}'$;
            $s_h \leftarrow s$;
        end
    end
    $T \leftarrow T \setminus \{s^{-1}_{h-tenure}\}$;
    $T \leftarrow T \cup \{s^{-1}_h\}$;
    if $V(\boldsymbol{\omega}'_h)$ is better than $V(\boldsymbol{\omega}^\star)$ then
        $\boldsymbol{\omega}^\star \leftarrow \boldsymbol{\omega}'_h$;
    end
    $h \leftarrow h + 1$;
end
```

The key element in a tabu search is forbidding certain moves, which prevents the procedure from returning to some recently visited local optimum. This is accomplished by entering the move reversals into the tabu lists, which are then used to stop attempted backtracking to recently visited local optima.

In a tabu search implementation towards solving a particular type of combinatorial optimisation problem, several of the method's building blocks require complete adaptation. The solution encoding, move types implemented, move encoding and tabu tenure must all be designed with careful consideration to the particular problem. This is one of the drawbacks of the tabu search method, but the same is true for metaheuristics in general.

The attribute-based approach

An alternative to recording move reversals in a tabu list is to record so-called *move attributes* instead. Attributes may, for example, be the arcs of a TSP tour that are omitted as the tabu search moves to another tour.

Each solution-specific move s from $\boldsymbol{\omega}$ to $\boldsymbol{\omega}'$ is associated with a set of g attributes, each generated by an *attribute function* $a_p(s, \boldsymbol{\omega})$, $p = 1, 2, \ldots, g$. The set of all attributes is denoted by E and each attribute is equipped with an *aspiration function value* $A(e)$, $e \in E$. The aspiration function $A(e)$ stores the best objective function value known to be obtainable from $\boldsymbol{\omega}'$. Whenever a move from $\boldsymbol{\omega}$ to $\boldsymbol{\omega}'$ is implemented, the aspiration function values for the g attributes of

Algorithm 5.2: Attribute based tabu search algorithm.

Input: A starting solution $\boldsymbol{\omega}$, a neighbourhood, denoted by $S(\boldsymbol{\omega})$, an objective function $V(\boldsymbol{\omega})$ defined for a problem $\mathcal{C}$, a termination bound, denoted by *bound*, attribute functions $a_p(s, \boldsymbol{\omega})$, $p = 1, 2, \ldots, g$, an aspiration function $A(e)$, $e \in E$, and a tabu list tenure for each tabu list, denoted by t_p. $T_{q(p)}$ points to the list in which attribute p should be stored.
Output: The best solution $\boldsymbol{\omega}^\star$ found to $\mathcal{C}$.

```
$\boldsymbol{\omega}^\star \leftarrow \boldsymbol{\omega}$;
$h \leftarrow 1$;
for $p = 1$ to $g$ do
    $T_p \leftarrow \emptyset$;
    $p \leftarrow p + 1$;
end
while $V(\boldsymbol{\omega}^\star)$ is not better than bound do
    trial_solution_set $\leftarrow \{s(\boldsymbol{\omega}) : s \in S(\boldsymbol{\omega})\}$;
    trial $\leftarrow$ tabu;
    while trial = tabu do
        $\boldsymbol{\omega}' \leftarrow s(\boldsymbol{\omega}) \in$ trial_solution_set that has the best $V(s(\boldsymbol{\omega}))$;
        trial $\leftarrow$ not_tabu;
        for $p = 1$ to $g$ do
            $e_p \leftarrow a_p(s, \boldsymbol{\omega})$;
            if $e_p \in T_p$ AND $V(\boldsymbol{\omega}')$ is not better than $A(e_p)$ then
                trial $\leftarrow$ tabu;
            end
            $p \leftarrow p + 1$;
        end
        if trial = tabu then
            trial_solution_set $\leftarrow$ trial_solution_set $\setminus \{s(\boldsymbol{\omega})\}$;
        else
            $\boldsymbol{\omega}'_h \leftarrow \boldsymbol{\omega}'$;
            for $p = 1$ to $g$ do
                $e_{p,h} \leftarrow e_p$;
            end
        end
    end
    for $p = 1$ to $g$ do
        $T_p \leftarrow T_p \setminus \{e_{p,h-t_p}\}$;
        $T_{q(p)} \leftarrow T_{q(p)} \cup \{e_{p,h}\}$;
        $A(e_{p,h}) \leftarrow$ The best among $\{A(e_{p,h}), V(\boldsymbol{\omega}), V(\boldsymbol{\omega}'_h)\}$;
    end
    if $V(\boldsymbol{\omega}'_h)$ is better than $V(\boldsymbol{\omega}^\star)$ then
        $\boldsymbol{\omega}^\star \leftarrow \boldsymbol{\omega}'_h$;
    end
    $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega}'_h$;
    $h \leftarrow h + 1$;
end
```

the move are updated according to $A(e) = \text{Min}\{A(e), V(\boldsymbol{\omega}), V(\boldsymbol{\omega}')\}$.

Each attribute is recorded in a *corresponding* tabu list, where the pointer $q(p)$ takes the index value of the tabu list in which attribute e_p should be stored. At iteration h the lists are

$$T_{q(p)} = \left\{a_p(s_k, \boldsymbol{\omega}_k) : k > h - t_{q(p)}\right\}, \qquad p = 1, 2, \ldots g, \tag{5.73}$$

where s_k and $\boldsymbol{\omega}_k$ refer to respectively the move s and the current solution $\boldsymbol{\omega}$ added during iteration k. The tabu lists (5.73) may be of different lengths t_p for different attributes.

A typical attribute-based tabu search procedure consists of completing the steps outlined in Algorithm 5.2.

When implementing a tabu search, a move may be allowed notwithstanding a non-passing status for one or more of the attributes. This may be allowed to occur if there are multiple types of attributes of varying importance. Sometimes a percentage of attributes having a passing status may suffice for a move to be allowed. In addition to this, when an attribute e has an aspiration function value $A(e)$ that is better than some predetermined value (often the best objective function value found so far during the search), e receives a passing status regardless of whether it is on a tabu list.

The approach for the TSP

The attribute-based tabu search method described above may be implemented towards solving the TSP. In this section it is described how the required input to Algorithm 5.2 is generated, if $\mathcal{C}$ is the TSP.

The solution encoding $\boldsymbol{\omega}$ may be taken to be a column vector whose elements are the vertices of a tour, so that $\boldsymbol{\omega}(j) = i$ indicates that vertex $i \in V$ is visited as the j^{th} vertex in the tour. A tour $\boldsymbol{\omega}$ may be generated either by means of some heuristic, or randomly (see, for example, the interesting discussion on how to generate an initial tour by DePuy *et al.* [50]). This encoded tour $\boldsymbol{\omega}$ constitutes the input starting solution in Algorithm 5.2.

A neighbourhood $S(\boldsymbol{\omega})$ for the TSP requires that a move type S be defined. The move type S is defined with respect to the solution encoding, and an example is to allow any two elements of $\boldsymbol{\omega}$ to *swap* positions. The new solution is denoted by $\boldsymbol{\omega}'$, as mentioned earlier, and a particular ordered pair of positions (j_1, j_2) in $\boldsymbol{\omega}$ may be denoted by the earlier defined move s, so that $s = (j_1, j_2)$ means that element $\boldsymbol{\omega}(j_1)$ moves to position j_2 in the tour and element $\boldsymbol{\omega}(j_2)$ moves to position j_1 in the tour. Another example is a *shift* move, in which one element is shifted from its current position to another position. Other move types exist, such as compound moves where a series of swaps or shifts are performed within the same "main" move. Yet other move types may be the shifting of subsets of vector entries at once, or the swapping of two subsets of entries at once. The more elements that are moved at the same time, the larger the *diversification* of the "main" move. In terms of the algorithmic implementation of the method, the neighbourhood is a function in which all possible moves implied by the move type and current solution are generated. Each move is then performed on the current solution, generating all possible trial solutions (or neighbours to the current solution), storing them as a set in a list or array. This neighbourhood constitutes the neighbourhood $S(\boldsymbol{\omega})$ in Algorithm 5.2.

The objective function of the TSP in terms of the solution encoding $\boldsymbol{\omega}$ sums the arc costs implied by the definition that each element in the encoding represents a vertex in a tour and that two elements that are adjacent (a neighbouring pair) in the encoding are also adjacent in the tour. Such a function constitutes the objective function $V(\boldsymbol{\omega})$ in Algorithm 5.2.

A good *bound* on the best possible value for the TSP's objective function value may be computed using the relaxation described in §5.2.1. This bound may be multiplied by some percentage, indicating to which degree the algorithm must fulfil this goal.

An attribute function within a tabu search implementation towards solving the TSP may return a so-called *neighbouring pair*, denoted by $e = (i, j)$, where $i, j \in V$ and $i \neq j$, which is defined as any two neighbouring vertices (in terms of their position in the solution encoding), together constituting a set denoted by E, the attribute set. Furthermore, the solution encoding $\boldsymbol{\omega}$ may be "closed" by "connecting" its last vertex with its first vertex (*i.e.* by wrapping the solution encoding). A solution sequence $\boldsymbol{\omega}$ of vertices may then be described by a subset of E constructed by including those neighbouring pairs which appear in $\boldsymbol{\omega}$. For example, if $\boldsymbol{\omega} = [2,3,1]$, the neighbouring pairs are $(2,3)$, $(3,1)$ and $(1,2)$, the rightmost neighbouring pair $(1,2)$ constituting the closing (or wrapping) of the sequence.

The move from a current solution to a trial solution may now be described in terms of *deleted* and *added* neighbouring pairs. For example, if moving from $\boldsymbol{\omega} = [2,3,1]$ to $\boldsymbol{\omega}' = [3,2,1]$ (by shifting vertex 3 into position 1) none of the three neighbouring pairs $(2,3)$, $(3,1)$ or $(1,2)$ remain neighbouring pairs in $\boldsymbol{\omega}'$, then these neighbouring pairs are said to have been deleted. The three added neighbouring pairs are $(3,2)$, $(2,1)$ and $(1,3)$. This example illustrates that for a TSP, a natural set of attribute functions consists of a first attribute function rendering the leftmost deleted neighbouring pair, a second attribute function rendering the middle deleted neighbouring pair, a third attribute function rendering the rightmost deleted neighbouring pair, a fourth attribute function rendering the leftmost added neighbouring pair, a fifth attribute function rendering the middle added neighbouring pair and a sixth attribute function rendering the rightmost added neighbouring pair. These attribute functions may constitute the $g = 6$ attribute functions $a_p(s,\boldsymbol{\omega})$, $p = 1,2,\ldots,g$, in Algorithm 5.2.

The aspiration function $A(e)$ in Algorithm 5.2 retains values as a function of neighbouring pair. A particular neighbouring pair, whether having been deleted or added, may be equipped with a value which is the best objective function value obtainable by deleting or adding it. Sometimes this function is replaced by the best objective function value found so far during the tabu search application (a single value for all neighbouring pairs in E).

5.3.2 Other approximate solution approaches

Some of the other approximate solution approaches which are commonly applied to the combinatorial optimisation problems outlined in §5.1 are briefly described in this section. However, none of these methods were implemented in the DSS presented later in this dissertation.

Some of these methods employ a special kind of search method known as *local search.* The local search method starts with a current solution. In a discrete setting (with respect to solution encoding), the local search method subsequently and in an iterative and strictly improving manner moves to better solutions in the neighbourhood defined by a predetermined move type until no better solution may be found, and then terminates. In a continuous setting, the search "steps" across the solution domain according to a predetermined step length, the direction being the direction of the gradient of the objective function evaluated at the current solution, until no better solution may be found along the gradient using the current step size, and then terminates.

Ant colony optimisation

Ant colony optimisation (ACO) was invented by Dorigo *et al.* [52] in 1996 and is a metaheuristic based on the notion that groups of ants are good at finding the shortest path between their colony and some source of food. The following metaphorical discussion is based on [52]. A colony consists of a large number of ants, each behaving in a certain manner. Each ant continuously deposits a chemical substance known to attract other ants, a so-called *pheromone.* The pheromone trail of an ant dissipates as time progresses. Each ant wanders more or less at random within a certain area, but is likely to follow any pheromone trail leading from one point to the next, the probability of which depends on the intensity of the pheromone trail. Thus, the more ants that follow a certain pheromone trail, the stronger the pheromone trail becomes, in turn increasing the probability of more ants following the pheromone trail.

A single ant is not likely to follow a particular pheromone trail for very long if only one other ant has recently deposited pheromone there. It is when an ant finds a food source, that the behaviour yields results. An ant that finds a food source stops and forages until it returns to the nest. When it returns, it is likely to return along the path by which it reached the food source, reinforcing the pheromone trail it deposited earlier. Other ants are likely to pick up the pheromone trail on the path that leads to the food source.

Since the world of ants is made up of large obstacles, ants must make decisions whether to go left or right around such obstacles at many points during their journeys. If, at some point, called A, on the path leading from the food source to the nest there exists an obstacle, the first ant is assumed to decide to go left or right around the obstacle with 0.5 probability assigned to either direction. Directly on the other side of the obstacle lies point B, where the split path around the obstacle is joined. If the left path takes twice as long to complete as the right path, an ant choosing the left path will reach point B later than an ant choosing the right path. Ants coming from the other direction will, until the first ant reaches point B, be equally likely to choose the left path or the right path. Assuming that the first ant choses the left path, an equal (probabilistically speaking) number of ants will take the right path as will take the left path until the first ant reaches point B. Then more ants will choose the left path for a while. However, ants on the right path towards the food source will reach the food source sooner than the other ants, thus returning sooner as well. When they reach point A, there will be a stronger pheromone trail on the right path since there were ants there more recently than on the left path. Ants are thus more likely to choose the shorter path since the shorter path is more likely to have recently been trodden.

Another manner in which to investigate this metaphor is to consider a steady-state situation *without* pheromones having been deposited. Assume that 10 ants in either direction per minute embark on the journey between the food source and the nest, and that the left path around the obstacle takes 10 minutes and the right path takes 5 minutes for the average ant to complete. The left path around the obstacle will then (probabilistically) be occupied by 50 ants moving in each direction, a total of 100 ants. The right path will be occupied by 50 ants in total[11]. This simple example shows that without the deposit of a pheromone trail, the ant density would be the same for the short and long paths. In fact, if the pheromone deposit and dissipation is "turned on" at this stage, the pheromone density would remain the same for the two paths indefinitely. One way for the pure pheromone metaphor to work is if the pheromone deposit and dissipation is turned on from the beginning, because of the initial head start that ants who finish the short route then give the pheromone trail on the short path. Another is to divide the whole path into sub-paths and for ants to deposit the same amount of pheromone per sub-path, thus causing a higher pheromone trail density on shorter sub-paths.

The ACO method of Dorigo *et al.* [52] is an iterative method. Suppose that a solution to a combinatorial optimisation problem $\mathcal{C}$ is encoded as a vector $\boldsymbol{\omega}$ of *components* ω_i, i indicating the identity of each component. The set of all components is denoted by V_{comp}. Let $\boldsymbol{\omega}(j) = \omega_i$ indicate that position j of the solution encoding vector contains component ω_i. The division into components must be performed in such a way as to ensure that the number of components required to complete a feasible encoding is constant within a given instance of $\mathcal{C}$.

In the ACO method, each component is associated with a *pheromone amount*, here denoted by $f_{\omega_i,k}$, where k is the current *outer* iteration of the method.

[11]In 10 minutes, 100 ants will have arrived at either point A or B, 50 deciding to follow the left path and 50 deciding to follow the right path, but in the same ten minutes half of the ants that entered the right path will have traversed and left it.

There are several *ants* in the ACO method, as in its related metaphor. These ants, denoted by $\boldsymbol{\omega}^u$, where u is an identifying index for the ants, each start an *inner iteration*, counted by an iteration counter ℓ, by containing a single starting component, and components are added in an iterative manner until $\boldsymbol{\omega}^u$ is feasible. At inner iteration ℓ and outer iteration k, a component ω_i has a certain desirability assigned to it, based on the value of $f_{\omega_i,k}$ and the cost of the component with respect to the objective function value of $\mathcal{C}$. Only components that have not yet been included and that cause the ant to retain its possibility of feasibility are eligible for selection. The selection is made probabilistically, based on each eligible component's desirability. For example, an eligible component with a desirability value of 3 is twice as likely to be selected as an eligible component with a desirability of 1.5. The set of eligible components changes between inner iterations, and towards the end of an outer iteration, is rather small. At the end of an outer iteration, the ant yields a feasible solution to $\mathcal{C}$, and $f_{\omega_i,k}$ is updated for all components. The components that were included in any ant receives more pheromone and all components have their pheromone amount decreased by some factor or function.

Algorithm 5.3 shows the inner and outer iteration loops with $V(\omega_i)$ denoting the objective function value change due to including component ω_i in any solution, $d^{\text{des}}(f_{\omega_i,k}, V(\omega_i))$ denoting the desirability function and ρ_f denoting the factor by which the pheromone present at each component is dissipated at each outer iteration.

Algorithm 5.3: Ant colony optimisation algorithm.

Input: Problem data and a bound on the maximum number of non-improving iterations *max_no_imprv_iter*.
Output: The best solution found to the problem ($\boldsymbol{\omega}^\star$).

```
$k \leftarrow 0$;
$f_{\omega_i,0} \leftarrow 0$ for all $\omega_i \in V_{\text{comp}}$;
$no\_imprv\_iter = 0$;
while $no\_imprv\_iter \leq max\_no\_imprv\_iter$ do
    $k \leftarrow k + 1$;
    Generate $m$ ants $\boldsymbol{\omega}^u$, $u = \{1, 2, \ldots, m\}$;
    Randomly assign a starting component to each ant;
    Update each ant's corresponding list of eligible components, denoted by $E_u^{ant}$;
    $\ell \leftarrow 0$;
    while $\ell$ is less than the number of $\omega_i$ required to complete a feasible solution do
        $\ell \leftarrow \ell + 1$;
        for $u = 1$ to $m$ do
            Select a component $\omega_i \in E_u^{ant}$ probabilistically with respect to $d^{\text{des}}(f_{\omega_i,k-1}, V(\omega_i))$;
            Add $\omega_i$ to $\boldsymbol{\omega}^u$;
            $E_u^{ant} \leftarrow E_u^{ant} \setminus \{\omega_i$ and other components which may no longer be added$\}$;
            $E_u^{ant} \leftarrow E_u^{ant} \cup \{$components which may now be added$\}$;
            $times\_added(\omega_i) \leftarrow times\_added(\omega_i) + 1$;
        end
    end
    Update the pheromone trail $f_{\omega_i,k}$ for all $\omega_i \in V_{\text{comp}}$ based on $times\_added(\omega_i)$;
    Let $times\_added = 0$ for all components;
    Dissipate the pheromone trail by letting $f_{\omega_i,k} \leftarrow \rho_f f_{\omega_i,k}$ for all $\omega_i \in V_{\text{comp}}$;
    if $V(\boldsymbol{\omega}^u)$ is better than $V(\boldsymbol{\omega}^\star)$ then
        $V(\boldsymbol{\omega}^\star) \leftarrow V(\boldsymbol{\omega}^u)$;
        $no\_imprv\_iter \leftarrow 0$;
    else
        $no\_imprv\_iter \leftarrow no\_imprv\_iter + 1$;
    end
end
```

Genetic algorithms

A *genetic algorithm* (GA) is special case of a more general algorithm known as an *evolutionary algorithm* [131]. Genetic algorithms were invented by John Holland in 1962 [104]. In the early days, the metaphor behind this class of algorithms consisted of terms such as *programs* representing a solution encoding and the term *environments* representing problems (see [104]), but now, after years of refining, the metaphor is more often described in the literature using a different setting.

In evolution, each species faces a continuously changing environment, be it the weather, resources, competition from other species or even competition from mutated individuals within the same species. These changes forces *natural selection* to occur, and the populations of individuals that comprise a species have certain means at their disposal by which they may evolve by adapting to the environment. One such evolution process occurs as individuals breed and give birth to a new individual, an individual that may or may not be fitter than its parents. If the individual is fitter than its competitors, it is likely to live longer, thus likely to breed more and to propagate its genes to the next generation. Another evolution process occurs when an individual mutates, *i.e.* experiences a small reproduction error in one or several of its genes. Usually this renders an unfit individual, but may also occasionally give rise to a superior individual whose new gene(s) will rapidly proliferate through breeding. Furthermore, a population of a species may encounter an entire foreign sub-population of the same species which has migrated to the first sub-population's territory, giving rise to an influx of new genes to the first sub-population. The new genes will be successful during the subsequent breeding and competition, if they combine with the genes of the individuals of the first sub-population to give rise to fitter individuals. The weakest individuals die from general weakness, and are more likely to succumb to disease, predators and competition from others within the same species. The fact that the fitter individuals are more likely to survive is known as *elitist selection*, and seemingly guarantees the survival of the species for generations to come.

The individuals may metaphorically be referred to as *chromosomes*, consisting of *genes.* Chromosomes comprise a *population.* A *generation* is a specific population in time, and time is represented by *iterations* so that one generation exists per iteration. The mutations, breeding, migrating populations and competition for resources and space, cause the next generation to adapt to its changing environment, its *basis for selection.*

A GA maintains a set of candidate solutions to a combinatorial optimisation problem and evolves it over multiple transformative steps, similarly to the above metaphor. The set of solutions is called the *current generation.* Several sub-procedures change solutions from the current population and add them to the *new generation*, thus advancing the current generation by one iteration. A pseudo-code description of the general approach is shown in Algorithm 5.4.

The solutions are often called chromosomes, and the sub-procedures have names such as *migration*, *cross-over*, *mutation* and *elitist selection.* Migration is a procedure that adds randomly generated chromosomes to the new population, while a cross-over is the combination of parts from two *parent* chromosomes from the current generation to form one or more *child* chromosomes to become part of the new generation. Mutation is implemented by taking a chromosome from the current generation and changing it randomly (or otherwise) so as to achieve *diversification* in the next generation. Elitist selection is the simple carrying over of some top proportion—in terms of objective function value—of the current generation to the new generation.

Algorithm 5.4: Genetic algorithm.

Input: Problem data and a bound on the maximum number of non-improving iterations *max_no_imprv_iter*.
Output: The best solution found to the problem (*best_solution*).

```
Generate initial current generation $P^{cur}$;
$no\_imprv\_iter = 0$;
while $no\_imprv\_iter \leq max\_no\_imprv\_iter$ do
    Generate new generation $P^{new}$ by
    {
      Performing generation on $P^{cur}$;
      Performing elitist selection on $P^{cur}$;
      Performing crossover on $P^{cur}$;
    };
    Evaluate $P^{new}$;
    if New best solution is found then
        $best\_solution$ = New best solution;
        $no\_imprv\_iter = 0$;
    else
        $no\_imprv\_iter = no\_imprv\_iter + 1$;
    end
    Set $P^{cur} = P^{new}$;
end
```

Memetic algorithms

Cheng and Gen [33] describe *memetic algorithms* (MAs) as hybrids between GAs and *local search*-based metaheuristics. Simply put, an MA is a GA in which the child chromosomes are added to the new generation only after having been improved by some local search-based metaheuristic. Recent attention in the combinatorial optimisation research literature and the general appeal of memetic algorithms probably stem from the combination of the diversifying strengths of a GA with the intensification strengths of local search-based heuristics. The general approach is described in pseudo-code in Algorithm 5.5.

Algorithm 5.5: Memetic algorithm.

Input: Problem data and a bound on the maximum number of non-improving iterations *max_no_imprv_iter*.
Output: The best solution found to the problem (*best_solution*).

```
Generate initial current population $P^{cur}$;
$no\_imprv\_iter = 0$;
while $no\_imprv\_iter \leq max\_no\_imprv\_iter$ do
    Generate intermediate new population $P^{prime}$ by
    {
      Performing migration on $P^{cur}$;
      Performing elitist selection on $P^{cur}$;
      Performing crossover on $P^{cur}$;
    };
    Improve each solution in $P^{prime}$ by means of a local search-based method;
    $P^{new} \leftarrow P^{prime}$ Evaluate $P^{new}$;
    if New best solution is found then
        $best\_solution$ = New best solution;
        $no\_imprv\_iter = 0$;
    else
        $no\_imprv\_iter = no\_imprv\_iter + 1$;
    end
    Set $P^{cur} = P^{new}$;
end
```

Scatter search

A *scatter search* (SS) is based on the notion of maintaining a *reference set* of good and diverse solutions. According to Belfiore [12], the working of a typical SS comprises the steps outlined in Algorithm 5.6. First, a starting solution set is generated by means of a diversifying sub-procedure, then each solution in this set is improved by means of a second sub-procedure. The first reference set of the SS is constructed by picking some of the best solutions and some of the most diverse solutions obtained via the set of improved solutions from the second sub-procedure.

The reference set is then subjected to three further sub-procedures until there is no change in its composition: solutions are combined, solutions are improved, and the reference set is updated. When these sub-procedures no longer cause any change in the reference set, it is rebuilt using other sub-procedures and an iteration counter is updated. During the SS, the best solution encountered is stored in memory. When the iteration counter reaches a certain value, the SS algorithm terminates.

In contrast to GAs, SSs perform each sub-procedure according to deterministic rules. These rules are highly involved and problem-specific and the interested reader is referred to the paper by Belfiore [12] where an SS for a heterogeneous fleet VRP with time windows and split deliveries is described.

Algorithm 5.6: Scatter search.

```
Input: Problem data and a bound on the maximum number of iterations $max\_iter$.
Output: The best solution found to the problem ($best\_solution$).
Generate an initial diverse solution set $P^{init}$;
$iter = 0$;
while $iter \leq max\_iter$ do
    Improve each solution in $P^{init}$ by means of, for example, a local search-based method;
    Select a set of good $P^{good}$ and a set of diverse $P^{div}$ solutions from $P^{init}$;
    $P^{ref} \leftarrow P^{good} \cup P^{div}$;
    while $P^{ref}$ changes do
        Combine solutions in $P^{ref}$ to form the set $P^{comb}$ ;
        Improve all solutions in $P^{comb}$;
        Select a set of good $P^{good}$ and a set of diverse $P^{div}$ solutions from $P^{comb}$;
        Update $P^{ref}$ by replacing certain solutions in $P^{ref}$ with solutions from $P^{good}$ and $P^{div}$;
    end
    if New best solution is found then
        $best\_solution$ = New best solution;
    end
    Rebuild $P^{ref}$;
end
```

Simulated annealing

Annealing is the physical process by which a heated substance, such as metal, alloy or glass, is allowed to cool gradually, in order for certain crystalline patterns among atoms to be allowed time to form before solidification of the substance occurs. In 1983, Kirkpatrick *et al.* [120] developed the idea to simulate this process for the purposes of solving combinatorial optimisation problems. In the method of *simulated annealing* (SA) a set of solutions is not maintained; moves from one solution to the next are applied instead, as is the case in the method of TS. As with TS, SA also requires a neighbourhood to be defined. From this neighbourhood, the algorithm chooses a new solution at random. Assuming that a combinatorial maximisation problem is considered, let δ^z denote the difference between the new objective function value and the current objective function value, and let T denote a value (conventionally called the "temperature") which decreases by a factor r $(0 \leq r \leq 1)$ each iteration. A positive value on δ^z indicates that the move being considered for acceptance is an improvement (increase) in terms of the objective function value. If so, the new solution is accepted and becomes the new current solution. If not, the new solution is accepted with probability $e^{-\delta^z T}$. While T is large, the probability of accepting a degrading solution is relatively high, which supposedly causes enough diversification for the method to be able to escape local optima early during the search. As T decreases, the algorithmic behaviour converges towards that of a hill-climbing heuristic that chooses the first available improving solution in the neighbourhood, therefore zooming in on a locally optimal solution. The method is described in the form of pseudo-code in Algorithm 5.7.

Algorithm 5.7: Simulated annealing.

Input: Problem data and a bound on the maximum number of non-improving iterations *max_no_imprv_iter*.
Output: The best solution found to the problem (*best_solution*).

```
Generate an initial solution $\boldsymbol{\omega}$;
$iter \leftarrow 0$;
while $no\_imprv\_iter \leq max\_no\_imprv\_iter$ do
    $\boldsymbol{\omega}' \leftarrow$ a solution in the neighbourhood of $\boldsymbol{\omega}$;
    if $\delta^z > 0$ then
        $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega}'$;
        $no\_imprv\_iter \leftarrow 0$;
    else if $[0,1]$-random number $< e^{-\delta^z T}$ then
        $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega}'$;
    else
        $no\_imprv\_iter \leftarrow no\_imprv\_iter + 1$;
    end
    if New best solution is found then
        $best\_solution \leftarrow$ New best solution;
    end
    $T \leftarrow rT$;
end
```

5.4 Practical aspects of the various solution methodologies

The only problem reviewed in §5.1 whose optimal solution may be found—or even approximated well—in a polynomial number of computational steps is the CAP [9, 125, 174]. The other problems described in §5.1 belong to a class of problems for which this is believed to be impossible [178], namely the class of *NP-hard problems*.

5.4.1 Solving the GAP in practice

Ross and Soland [176] designed a branch-and-bound algorithm for the GAP in 1975 which was capable, at the time, of solving instances with 5, 10 or 20 agents and 200 jobs in consistently under 3 seconds in FORTRAN IV on a CDC 6600 with 72 000 words of memory. They used a branch-and-bound approach in which the *Lagrangean relaxation* (see, for example, [56]) consisted of allowing jobs to be assigned to agents without the restriction of limited resources (*i.e.* relaxing (5.6)). An optimal solution to the relaxation is easily found by assigning each job to an agent with lowest cost. A lower bound on the objective function value of an optimal solution to the original problem is thus achieved, allowing for an optimality-related stopping rule. Ross and Soland moreover refined this lower bound by identifying those constraints in the original problem that were violated by a solution to the relaxation. A comparable branch-and-bound-based algorithm is the one of Martello and Toth [146], presented in 1981, in which the constraint set ensuring that each job is only assigned to any agent once—*i.e.* (5.7)—is relaxed using Lagrangean relaxation, which breaks up the GAP into an equal number of *knapsack problems* as there are agents in the GAP. In terms of the lower bound achieved when using Lagrangean relaxation, a part of the relaxation is penalised by means of Lagrangean multipliers for each relaxed constraint. Later, in 1986, Fisher *et al.* [57] also relaxed the constraint set (5.7) in their branch-and-bound-based algorithm, but used a different method than Martello and Toth by which to compute the Lagrangean multipliers. At the time, these three algorithms were evidently the best for solving the GAP.

According to more recent computational comparisons by Yagiura *et al.* [238] in 2004, the branch-and-bound-based exact method of Nauss [161] performed on par with their tabu search-based metaheuristic on difficult (class-E)[12] instances with up to 20 agents and 200 jobs. In [238], all instances with more than 300 jobs were only attempted with metaheuristics, and the largest class-E instances had 1 600 jobs and 80 agents. Such problems were attempted for up to 50 000 seconds yielding solutions within 0.03 % from the lower bound. Racer and Amini [174] employed a *variable depth search*-based algorithm and Higgins [95] adopted a *tabu search* solution approach on sugarcane harvest scheduling problems with 50 000 jobs and 40 agents. The currently most powerful algorithms for the GAP appear to be those in [51, 238, 239].

5.4.2 Solving the asymmetric/symmetric TSP in practice

As mentioned above, Dantzig *et al.* [45] adopted a mathematical programming approach towards solving the TSP, based on a linear programming relaxation, and in doing so, developed a method that still forms the basis for some of the best exact algorithms presently available. The symmetric TSP belongs to the class of *NP-hard* problems [39] and has been under investigation by the modern research community for more than 50 years. Grötschel and Holland [82], as well as Padberg and Rinaldi [166], developed the methods of Dantzig *et al.* further in two branch-and-cut implementations, the latter of which forms the core of the Concorde TSP by Applegate *et al.* [4], possibly the best exact algorithm currently available for the TSP. According to Helsgaun [91], the ATSP is usually less difficult to solve than the symmetric TSP when using exact algorithms. Miller and Pekny [154] reported a 500 000 vertex ATSP instance being solved to optimality. Mak and Boland [145] presented a Lagrangean relaxation-based branch-and-bound algorithm that solves an ATSP instance with replenishment arcs with 519 vertices and 42449 arcs in 894 seconds, while CPLEX 9.0 (which employs a branch-and-bound approach) required a computational time of 1 144 seconds to solve the same instance, both computations performed on a Dell Latitude C840, Model PP01X with a P4 2 GHz CPU and unspecified RAM. Grötschel and Holland [82] solved 1 000-city symmetric TSPs in approximately 2 hours of CPU time on an IBM 3081D with a 38 MHz CPU and 16 Mb RAM.

A metaheuristic based on the groundbreaking work of Lin and Kernighan during the 1970s [141] and named LKH 2 by its author, Helsgaun [92], solved an 85 900 city instance of a symmetric TSP to optimality, a 1 000 000 city instance to within 0.027 % of optimality and a 10 000 000 city instance to within 0.58 % of optimality, all in 2009. The optimal solution of the Helsgaun 85 900 city instance was established by Applegate *et al.* [4].

[12]In mainstream GAP computational comparisons, difficult instances are often called E-instances following a well-established convention of problem instance generation.

Further approximate approaches towards solving the symmetric TSP include a hybrid genetic algorithm by Jayalakshmi and Sathiamoorthy [115] and a metaheuristic *randomised priority search* by DePuy *et al.* [50]. A genetic algorithm for the ASTP of Choi *et al.* [37] performed, according to their own computational comparisons, no better than the branch-and-bound-based algorithm of Carpaneto *et al.* [29]. Xing *et al.* [237] presented a hybrid GA which produces high quality solutions to the ATSP with long computing times.

5.4.3 Solving time-dependent ATSPs and TSPs in practice

The ATSPTDC may, according to Albiach *et al.* [5], be transformed into the ATSP and subsequently solved using ATSP algorithms or further transformed and solved using TSP algorithms. This may be a highly viable option, since the amount of research available on exact algorithms for the ATSP or TSP is vast compared with that available for the time-dependent problems in §5.1. Consider, however, that the state-of-the-art work in [5] only manages to solve ATSPTDCs with 60 vertices and time window-widths of 30 instants in approximately one to three hours of CPU time on a PC with 1.8 GHz CPU and unspecified RAM. When the number of time instants at each of the 60 vertices was equal to 60, no optimal solutions could be found.

The VRP with the addition of time windows becomes very similar to the ATSPTW if the number of vehicles is reduced to one. Kohl and Madsen [122] solved many instances with up to 100 customers to optimality using an exact approach based on Lagrangian relaxation and the branch-and-bound method. However, these problems are not time-dependent in the parameters; they merely have time-independent travelling times and travelling costs[13]. A computational study of a memetic algorithm by Nagata *et al.* [160] included attempts to solve instances as large as 1 000 customers for the VRP with time windows.

Stecco *et al.* [207, p. 14] applied a tabu search to solve an STDSP similar to the TDTSP, claiming to produce better and faster results than the available exact algorithms for all instances for which the exact algorithms did not find an optimal solution (in this case, the exact approaches were stopped before finding an optimal solution). The tabu search produced very good solutions to instances with 50 jobs in less than 2 minutes on a PC with a 3.0 GHz CPU and 1 Gb of RAM, while the same instances often required 240 minutes in their branch-and-bound approach (on the same PC), as described in [206].

It seems that the choice between a metaheuristic and an exact algorithm should fall on the metaheuristic when interest is purely practical and the problem size is moderate to large (*i.e.* 50 or more cities/jobs), and on the exact algorithm when the problem size is smaller and the interest is more theoretical. One may, however, often benefit from formulating a combinatorial optimisation problem as an integer or mixed integer linear programming problem so that a lower or upper bound may be computed by means of its linear programming relaxation. Alternatively, a lower or upper bound may be found by other forms of relaxation, such as removing certain sets of constraints, thus transforming the problem into one more easily solved.

5.5 Chapter summary

A number of classical problems in the operations research literature were briefly reviewed in this chapter, in partial fulfilment of Dissertation Objective II, as stated in §1.3. Each classical

[13]This causes a reduction in the number of constraints, since the time accounting may be completed by means of variables rather than ensured through the addition of constraints.

problem was conceptually connected to the THSP, in an effort to highlight the relevance of these problems towards modelling the THSP. An attempt was made to formulate the ATSPTDC as an integer programming problem, such a formulation not having been encountered during the literature review.

The branch-and-bound method and the method of cutting planes were described in §5.2 as exact solution methodologies for the problems in §5.1, in partial fulfilment of Dissertation Objective II, as stated in §1.3. These two methods are, in fact, the starting points for most exact approaches towards solving *NP-hard* combinatorial optimisation problems. It is unlikely, however, that very large instances to some of these problems will be solved to optimality in the near future, considering that the research community regards time-dependent TSPs with 100 nodes to be large with respect to current computing capabilities and current computing technology.

A number of popular metaheuristics were briefly reviewed in §5.3, in partial fulfilment of Dissertation Objective II, as stated in §1.3. These include ant colony optimisation, genetic algorithms, memetic algorithms, scatter search, simulated annealing and tabu search. These methods show the ability to uncover very good solutions to very large, practical size instances of the problems in §5.1.

Finally, some practical aspects of the various solution methodologies were discussed in §5.4, completing the fulfilment of Dissertation Objective II, as stated in §1.3. The choice was made to design the core of the DSS presented later in this dissertation based on an alternative formulation of the ATSPTDC to the one presented in §5.1.7 coupled with a tabu search metaheuristic similar to the one presented in §5.3.1.

CHAPTER 6

Harvest scheduling models

Contents

An early version of the DSS put forward in this dissertation was based on a natural model for the THSP, introduced in §5.1.2 as the GAP. The resulting DSS-*development version* (DSSDV) was built entirely around the GAP-model. Its databases were implemented in Excel, and a purpose-built local search based solution procedure was implemented in Wolfram's Mathematica, the Excel implementation employing various schedule printing preparation macros as well. This chapter opens with a description of the optimisation model incorporated into the DSSDV in §6.1, and then continues with an outline of the local search-based solution approach.

A further DSS was developed incorporating improvements as a result of a validation experiment performed on the DSSDV, as will be described later in this dissertation. The optimisation model embedded in the final DSS is then described in §6.2, and this is followed by a description of the solution approach taken towards solving it. The two separate decision support systems consist of two conceptually different approaches in terms of the combinatorial optimisation models incorporated in them. The approach forming the basis for the final DSS is considered more suitable with respect to the THSP as will be argued later in this dissertation.

6.1 The base model formulation

The mathematical programming model incorporated into the DSSDV in order to generate seasonal harvesting schedules is presented in this section. The main purpose of this mathematical programming model is to decide during which time period to harvest each of the sugarcane fields in the area under consideration. The formulation allows for an arbitrary number of fields and an arbitrary number of time periods, and is based on the GAP, presented in §5.1.2. The GAP is augmented here by introducing a lower bound in the form of a constraint on the amount of "resource" used by each agent. At this developmental stage, most of the modelling effort was focused on formulating a sensible, uncomplicated model with little attempt at incorporating constraints for every possible eventuality. The GAP with the lower bound is henceforth referred to as the *base model formulation* (BMF), the word *base* referring to the fact that it formed the basis for further model development as will become clear later in this dissertation.

6.1.1 Integer programming model formulation

Let I be the set of fields for which a harvesting schedule is sought and let J be the set of harvesting periods over which a harvesting schedule is to be constructed. The length of a harvesting period may be any period of time that is sensible to adopt as a planning period, such as a week or a month. Furthermore, define the binary decision variable x_{ij} to take the value 1 if field i is harvested during period j, or 0 otherwise. Also, let the parameters P_{ij} and M_{ij} represent the profit forecast and the yield forecast associated with harvesting field i during period j, respectively. Finally, let D_j^{min} and D_j^{max} be the minimum and maximum tonnages allowed to be harvested during period j, respectively. Then the objective in the BMF is to

$$\text{maximise} \quad z = \sum_{i \in I} \sum_{j \in J} P_{ij} x_{ij} \tag{6.1}$$

subject to the constraints

$$\sum_{j \in J} x_{ij} = 1, \qquad i \in I, \tag{6.2}$$

$$\sum_{i \in I} x_{ij} M_{ij} \geq D_j^{min}, \qquad j \in J, \tag{6.3}$$

$$\sum_{i \in I} x_{ij} M_{ij} \leq D_j^{max}, \qquad j \in J, \tag{6.4}$$

$$x_{ij} \in \{0, 1\}, \qquad i \in I, \quad j \in J. \tag{6.5}$$

The constraint set (6.2) ensures that all fields are harvested exactly once, while constraint sets (6.3) and (6.4) impose respectively lower and upper bounds on the harvested tonnage during any period. Constraint set (6.5) ensures that entire fields are harvested (*i.e.* harvesting fractions of fields is disallowed). The formulation without constraint set (6.3) is equivalent to the traditional formulation of the GAP.

A major drawback of the BMF is its misrepresentation of the fact that during almost every period, one field is only partially harvested in practice. If, for instance, four fields have been scheduled to be harvested during a particular period, the first three are typically burnt and harvested in succession and the fourth is then burnt and may only be harvested in part due to the harvesting period coming to an end before its harvesting can be completed. The remaining part is often harvested during the following period. Harvesting schedules generated by the BMF do not exploit this opportunity for letting one field in each period be harvested partly, allowing for the completion of its harvesting to take place during the next period. In fact, it is probable that the real-world value of the schedules produced by the BMF are significantly diminished due to this drawback. Theoretically, the BMF may however be better at modelling harvesting scenarios where operations cease on the last one or two days of each week, since harvesting a burnt field only partly causes sucrose deterioration to occur in the remaining cane.

The fields must be harvested according to the DRD (see §3 and (3.1)), and hence each period must be assigned a set of fields which, when combined, fulfil the DRD during that period. The variability in the ability of a harvesting operation to fulfil the DRD is accommodated by the gap between (6.3) and (6.4). This gap is necessary for feasibility reasons as well, since a gap of 0 would mean that each period would have to be filled exactly by harvesting activity, which is not possible in a practical application. A breakdown of the THSP into periods was done while keeping in mind that the schedule is likely to change both *within* every period and *between* every two consecutive periods due to forced implementation deviations from the schedule. Conceptual restrictions foreseen by the author is that the model will not apply when the number of different business entities that share the same schedule increases. For example, if two hundred growers submit all their field records and *estimates*[1], and are later subjected to a grand schedule that outlines every field's harvest week, the model solution would probably not be adopted by the growers due to a variety of equity issues that are not explicitly taken into consideration, such as growers requiring cash-flow every month. Furthermore, it may not apply when the number of fields is too small, due to the combinatorial nature of the allocation of fields to time periods. For example, if three of the fields should be harvested first for reasons of profitability, the BMF may sometimes counteract this due to the fields not satisfying the first period's constraints in (6.3) and (6.4). The model is else foreseen to be applicable whenever a decision maker is considering when to harvest his or her sugarcane fields, or when a business is making such decisions for an entire set of farms or a large estate. The supply chain into which the farm(s) deliver may be of consequence; for example, transport costs may be poorly estimated by an average in some cases. If one mill is further away, it will rarely be profitable to deliver to it unless cane would be eligible to carry over (not be harvested until the next season). The DRD may easily be computed regardless of whether a farm is part of a single or several supply chains.

6.1.2 A local search solution approach

As part of the DSSDV, a solution approach which could be relied upon to consistently solve problem instances of up to 100 fields and 40 time periods had to be developed in rather a short time. The solution of the BMF was initially approached by means of two different methods. Firstly, a purpose-built local search procedure, here called the *BMF algorithm*, was coded in Wolfram's Mathematica [149] and secondly, Lindo System's LINGO 9 [140] was applied as a modelling language and solver to (6.1)–(6.5). LINGO 9 was unable to produce feasible solutions to practical size instances. Instead, the local search-based metaheuristic algorithm was employed. The main traits of the algorithm are described by means of pseudo-code shown in Algorithms 6.1–6.4. Later, it was found that LINGO 11 produced good feasible solutions relatively quickly to some problem instances, but this finding occurred too late to pose an alternative solution method.

For the purposes of describing the metaheuristic referred to above, a solution to the BMF is denoted by the vector $\boldsymbol{\sigma}$. The i^{th} element of $\boldsymbol{\sigma}$ is the number of the period during which field i is harvested. Therefore, $\boldsymbol{\sigma}(i) = j$ means that field i is harvested during period j. Let $V(\boldsymbol{\sigma})$ denote the objective function value of a solution $\boldsymbol{\sigma}$. Furthermore, let $\boldsymbol{\sigma}^\star$ denote the best solution found so far during the search.

In the BMF algorithm, shown in Algorithm 6.1, the problem (6.1)–(6.5) is first relaxed to a linear programming problem by modifying the binary constraint set (6.5) into the interval constraint set

$$0 \leq x_{ij} \leq 1, \qquad i \in I, \quad j \in J. \tag{6.6}$$

[1]The list containing all fields to be harvested during the current season.

Algorithm 6.1: BMF algorithm.

Input: Profit parameters P_{ij}, cane yield parameters M_{ij}, minimum cane yield D_j^{min}, maximum cane yield D_j^{max}, access to the current system time, denoted by *system_time*, and a time limit, denoted by *time_limit*.
Output: The best available allocation of fields to harvesting periods, denoted by $\boldsymbol{\sigma}^\star$.

```
$start\_time \leftarrow system\_time$;
$best \leftarrow -\infty$;
Compute an upper bound, denoted by $UB$, using a linear programming relaxation;
while $system\_time \leq time\_limit + start\_time$ do
    $\boldsymbol{\sigma} \leftarrow N_{\text{init\_shift}}(\boldsymbol{\sigma})$;
    while $iter < max\_iter$ do
        $best\_in \leftarrow best$;
        $\boldsymbol{\sigma} \leftarrow N_{\text{perturb}}(\boldsymbol{\sigma})$;
        $\boldsymbol{\sigma} \leftarrow N_{\text{shift}}(\boldsymbol{\sigma})$;
        if $best > best\_in$ then
            $iter \leftarrow 1$;
        else
            $iter \leftarrow iter + 1$;
        end
    end
end
```

Algorithm 6.2: Initial shift move $N_{\text{init\_shift}}(\boldsymbol{\sigma})$.

Input: The best objective function value found so far, denoted by *best*.
Output: A candidate solution $\boldsymbol{\sigma}$, and, if discovered, a new best solution $\boldsymbol{\sigma}^\star$.

```
Generate new random starting solution $\boldsymbol{\sigma}$;
if $V(\boldsymbol{\sigma}) > best$ then
    $\boldsymbol{\sigma}^\star \leftarrow \boldsymbol{\sigma}$;
end
$starter \leftarrow 1$;
while $starter = 1$ do
    $i \leftarrow 1$;
    $j \leftarrow 1$;
    $stopper \leftarrow 0$;
    while $stopper = 0$ do
        $\boldsymbol{\sigma}' \leftarrow \boldsymbol{\sigma}$;
        $\boldsymbol{\sigma}'(i) \leftarrow j$;
        $i \leftarrow i + 1$;
        if $i > |I|$ then
            $j \leftarrow j + 1$;
            $i \leftarrow 1$;
        end
        if $V(\boldsymbol{\sigma}') \geq V(\boldsymbol{\sigma})$ then
            $\boldsymbol{\sigma} \leftarrow \boldsymbol{\sigma}'$;
            $stopper \leftarrow 1$;
        end
        if $j > |J|$ then
            $stopper \leftarrow 1$;
            $starter \leftarrow 0$;
        end
    end
end
if $V(\boldsymbol{\sigma}) > best$ then
    $\boldsymbol{\sigma}^\star \leftarrow \boldsymbol{\sigma}$;
    $best \leftarrow V(\boldsymbol{\sigma}^\star)$;
end
```

The resulting linear programming problem is then solved by means of a function included as a standard function within Mathematica and the objective function value of its solution is taken as an upper bound on the optimal value of the objective function value in (6.1).

The BMF algorithm then applies a shift move function, denoted by $N_{\text{init\_shift}}(\boldsymbol{\sigma})$, which generates a starting solution for itself, improves that solution and returns the improved solution as output. This solution is passed on to a move function designed to diversify the search, denoted by

Algorithm 6.3: Perturbation move $N_{\text{perturb}}(\boldsymbol{\sigma})$.

Input: A solution $\boldsymbol{\sigma}$ and a *candidate ordering* $\tilde{\mathbf{c}}$ of the fields.
Output: A candidate solution $\boldsymbol{\sigma}$, and, if discovered, a new best solution $\boldsymbol{\sigma}^\star$.

```
while Not all fields have been selected to be the first ejected field do
    Select a field in $\boldsymbol{\sigma}$ according to $\tilde{\mathbf{c}}$ to be the first ejected field, denoted by ejf, and eject it from its period;
    Compute available DRD, denoted by availm, during the period that ejf was ejected from;
    while Not all fields have been examined do
        Select a field ik in $\boldsymbol{\sigma}$ according to $\tilde{\mathbf{c}}$ to potentially be ejected and subsequently move to the period that the previously ejected field occupied;
        if Mass of ik $\leq$ availm then
            Perform the move of ik in $\boldsymbol{\sigma}$;
            Compute availm during the period in $\boldsymbol{\sigma}$ that ik was ejected from;
            Create a trial solution by allowing ejk to move to the period that the previously ejected field occupied;
            Save the trial solution in a list of trial solutions;
        end
    end
    if best trial solution is better than $\boldsymbol{\sigma}$ then
        Let $\boldsymbol{\sigma}$ be the best trial solution;
    end
    if $V(\boldsymbol{\sigma}) > best$ then
        $\boldsymbol{\sigma}^\star \leftarrow \boldsymbol{\sigma}$;
        $best \leftarrow V(\boldsymbol{\sigma}^\star)$;
    end
end
```

$N_{\text{perturb}}(\boldsymbol{\sigma})$, which attempts to move fields around in $\boldsymbol{\sigma}$ while ensuring that the solution remains feasible. The third function employed is another a shift move function, denoted by $N_{\text{shift}}(\boldsymbol{\sigma})$.

The move function $N_{\text{init\_shift}}(\boldsymbol{\sigma})$ improves, if possible, the starting solution by repeatedly moving through the solution space by "shifting" fields from one harvesting period to another. A move is performed as soon as a "shift" is found that causes an improved objective function value.

Algorithm 6.4: Shift move $N_{\text{shift}}(\boldsymbol{\sigma})$.

Input: A starting solution $\boldsymbol{\sigma}$.
Output: A new solution $\boldsymbol{\sigma}$, and, if discovered, a new best solution $\boldsymbol{\sigma}^\star$.

```
$starter \leftarrow 1$;
while $starter = 1$ do
    $i \leftarrow 1$;
    $j \leftarrow 1$;
    $stopper \leftarrow 0$;
    while $stopper = 0$ do
        $\boldsymbol{\sigma}' \leftarrow \boldsymbol{\sigma}$;
        $\boldsymbol{\sigma}'(i) \leftarrow j$;
        $i \leftarrow i + 1$;
        if $i > |I|$ then
            $j \leftarrow j + 1$;
            $i \leftarrow 1$;
        end
        if $V(\boldsymbol{\sigma}') \geq V(\boldsymbol{\sigma})$ then
            $\boldsymbol{\sigma} \leftarrow \boldsymbol{\sigma}'$;
            $stopper \leftarrow 1$;
        end
        if $j > |J|$ then
            $stopper \leftarrow 1$;
            $starter \leftarrow 0$;
        end
    end
end
if $V(\boldsymbol{\sigma}) > best$ then
    $\boldsymbol{\sigma}^\star \leftarrow \boldsymbol{\sigma}$;
    $best \leftarrow V(\boldsymbol{\sigma}^\star)$;
end
```

Date	Problem size (Fields $\times$ Periods)	Time limit [s]	% Gap from bound
18 March 2009	36×19	3600	2.363
18 March 2009	72×19	3600	1.479
25 March 2009	72×18	7200	1.371
30 March 2009	33×18	7200	4.044
30 March 2009	72×18	7200	1.443
14 April 2009	33×17	1800	1.780
14 April 2009	72×17	7200	1.352
27 April 2009	32×16	7200	1.141
27 April 2009	70×16	7200	1.221
10 May 2009	27×15	1800	27.11
10 May 2009	58×15	5400	1.468
11 June 2009	52×13	7200	238.9
22 June 2009	24×12	60	8.157
22 June 2009	49×12	1800	21.66
10 July 2009	23×11	1800	0.842
10 July 2009	44×11	3600	1.996
1 August 2009	21×9	1800	0.716
1 August 2009	40×9	1800	1.981
10 August 2009	17×8	180	0.595
10 August 2009	38×8	180	1.259
25 August 2009	20×7	1800	0.524
25 August 2009	35×7	3600	0.805

Table 6.1: *Computational results obtained via the BMF algorithm during a validation experiment performed during the 2009 harvesting season together with a harvesting group of four medium-scale growers in the Eston Mill area in KwaZulu-Natal.*

This is known as a *first admissible move strategy.* The function returns the best solution it encounters, and, if applicable, updates the best solution found so far during application of the BMF algorithm. The procedure is described further by means of pseudo code in Algorithm 6.2.

The $N_{\text{perturb}}(\boldsymbol{\sigma})$ move function selects a "first" field, according to some *candidate ordering*, to be ejected from its harvesting period. Upon passing a test, a "second" field is moved into this harvesting period, a "third" field into the "second" field's period, and so on, until all fields have been tried for such a move. Each time a field has been moved, the first field is inserted into the last harvesting period to be "left" by a field, generating a so-called *trial solution*, which is stored in memory.

The $N_{\text{perturb}}(\boldsymbol{\sigma})$ move function then adopts the best trial solution as $\boldsymbol{\sigma}$ (if it is better than the previous value of $\boldsymbol{\sigma}$), and selects the next "first" field according to the *candidate ordering* and repeats the above procedure, continuing until all fields have played the part as "first" field. The main procedural workings of this function are shown in Algorithm 6.3.

The $N_{\text{shift}}(\boldsymbol{\sigma})$ move function is identical to the $N_{\text{init\_shift}}(\boldsymbol{\sigma})$ move function, except that $N_{\text{shift}}(\boldsymbol{\sigma})$ does not generate a new starting solution. The function is, however, shown in Algorithm 6.4.

The computational results obtained via the BMF are illustrated here by means of twenty-two actual runs during the 2009 harvesting season, based on data relevant to the Eston Mill area in KwaZulu-Natal. These runs were computed in Mathematica 6, installed on a 1.7 GHz PC with 1 GB RAM. The results are presented in Table 6.1 and illustrate problems in obtaining good optimality values encountered mainly when running small problems (with either a small number of fields or a small number of periods, or both).

6.2 The sequential model formulation

The mathematical programming model and solution methodology employed in the final DSS presented later in this dissertation, is fundamentally different from the BMF and is similar to the ATSPTDC, presented in §5.1.7. This approach is concerned with arranging the fields in a sequence, rather than into groups to be assigned to harvesting periods, as was the case in the BMF. This formulation is therefore called the *sequential model formulation* (SMF). The model formulation incorporates time- and sequence-dependent costs so that the time instant at which a field is harvested depends on the sum of the harvesting times of the preceding fields.

6.2.1 Integer programming model formulation

Let $I = \{0, 1, 2, \ldots, n, n+1\}$ be the set of fields that have to be harvested during the harvesting season. Here the fields 0 and $n+1$ are a dummy starting field and a dummy ending field, respectively, delimiting a feasible harvesting sequence. The combined time required to harvest field $u \in I$ and to physically travel to any other field is denoted by t_u. Let $J = \{1, 2, \ldots, b_0\}$ be the set of time instants (for example, days) into which the season is divided, where $b_0 = \sum_{u \in I} t_u$. Denote the profit from harvesting field $u \in I$ beginning at time instant $j \in J$ by P_{uj}. The parameters P_{0j}, $P_{n+1,j}$, t_0 and t_{n+1} are assumed to be zero for all $j \in J$. The decision variable x^δ_{uvj} is defined to take the value 1 if the harvesting operation leaves field $u \in I$ for field $v \in I$ at time instant $j \in J$, or the value 0 otherwise. Furthermore, let y_u denote the time instant at which the harvesting operation leaves field u and let $\mathbb{Z}^+$ denote the set of positive integers. The travelling times between fields are assumed to be negligible and the time required to harvest a field is assumed to be independent of the time of harvest. The objective of the SMF is then to

$$\text{maximise} \qquad z = \sum_{u \in I} \sum_{v \in I} \sum_{j \in J} P_{uj} x^\delta_{uvj} \tag{6.7}$$

subject to the constraints

$$\sum_{v \in I} \sum_{j \in J} x^\delta_{uvj} = 1, \qquad u \in I \setminus \{n+1\}, \tag{6.8}$$

$$\sum_{u \in I} \sum_{j \in J} x^\delta_{uvj} = 1, \qquad v \in I \setminus \{0\}, \tag{6.9}$$

$$\sum_{v \in I} \sum_{j \in J} x^\delta_{n+1,v,j} = 0, \tag{6.10}$$

$$\sum_{u \in I} \sum_{j \in J} x^\delta_{u0j} = 0, \tag{6.11}$$

$$x^\delta_{uvj} = 0, \qquad u = v \in I, \quad j \in J, \tag{6.12}$$

$$x^\delta_{0,n+1,j} = 0, \qquad j \in J, \tag{6.13}$$

$$y_0 = 1, \tag{6.14}$$

$$y_{n+1} - 1 = b_0, \tag{6.15}$$

$$y_u + t_u - M\left(1 - x^{\delta}_{uvj}\right) \leq y_v, \qquad u \in I \setminus \{n+1\}, \quad v \in I \setminus \{0\}, \quad j \in J, \tag{6.16}$$

$$\sum_{v \in I}\sum_{j \in J} j x^{\delta}_{uvj} = y_u, \qquad u \in I \setminus \{n+1\}, \tag{6.17}$$

$$x^{\delta}_{uvj} \in \{0,1\}, \qquad u, v \in I, \quad j \in J, \tag{6.18}$$

$$y_u \in \mathbb{Z}^+, \qquad u \in I, \tag{6.19}$$

where (6.7) computes the total harvesting operational profit, (6.8) ensures that all fields except the dummy ending field are "exited" exactly once in the harvesting sequence and (6.9) ensures that all fields except the dummy starting field are "entered" exactly once in the harvesting sequence. Constraint sets (6.10) and (6.11) ensure that the dummy ending field is not "exited" and that the dummy starting field is not "entered", (6.12) forbids looping from a field to itself in the harvesting sequence, while (6.13) ensures that the ending field is not "visited" immediately after the starting field in the harvesting sequence. Constraint sets (6.14), (6.15), (6.16) and (6.17) ensure that the time of harvest is advanced by t_u time instants when the harvesting operation moves from field u to field v. Finally, (6.18) ensures that the decision variable x^{δ}_{uvj} is binary and (6.19) ensures that the decision variable y_u is a positive integer.

In order to explore the ease with which the SMF may be solved, several pseudo-randomly generated instances of the SMF were solved in Lindo System's LINGO 11 [140] on an Intel Core2 vPro 3 GHz PC processor with 4 Gb RAM. LINGO 11 mainly employs a branch-and-bound solution approach but also a *relaxation induced neighbourhood search* (RINS), a metaheuristic designed to bring the advantages of local search to the realm of integer programming and mixed integer programming solution methodology (see [43] for a detailed description of RINS). The sizes of these problem instances and their solution times are shown in Table 6.2. In solving instance D, LINGO found an optimal solution within two hours, while the last six hours were spent improving the lower bound so as to establish optimality. No feasible solutions were found within fifteen hours for instance E. As a practical size problem instance may involve more than fifty fields and two hundred time instants, it became apparent that an alternative solution approach was required.

6.2.2 A tabu search solution approach

The *tabu search* (TS) approach was therefore adopted to solve the model (6.7)–(6.19). This local search is primarily based on the work by Glover [73] and inspired further by the paper of Yagiura *et al.* [238], which actually focuses on the GAP, but the solution method shares common features with the one presented here. The TS is shown by means of pseudo-code in Algorithm 6.5.

Instance	Problem size (Fields × Time instants)	CPU time [s]	Feasible solution?	Optimal solution?
A	5×14	3	Yes	Yes
B	5×28	15	Yes	Yes
C	7×21	30	Yes	Yes
D	8×21	29100	Yes	Yes
E	10×21	54000	No	—

Table 6.2: *Various SMF instances solved using* LINGO *11. The time unit is seconds.*

Algorithm 6.5: Attribute-based tabu search algorithm for the SMF.

Input: The objective function $z(\boldsymbol{\omega})$; a time limit bound *time_limit*; attribute functions $a_p(s, \boldsymbol{\omega})$, $p = 1, 2, \dots, g$; an aspiration function $A(e)$, $e \in E$; tabu list tenures t_p; pointers $q(p)$ to the list $T_{q(p)}$ in which attribute p should be stored. The shift move type $N^{SMF}_{\text{shift}}(\boldsymbol{\omega})$, the ejection chain compound move type $N^{SMF}_{\text{ejec}}(\boldsymbol{\omega})$ and the pseudo-random generation algorithmN^{SMF}_{RR}. The maximum number of non-improving iterations *max_no_local_impr* allowed between applications of the N^{SMF}_{ejec}. The maximum number of non-improving iterations *max_no_global_impr* allowed between applications of N^{SMF}_{RR}.

Output: The best solution $\boldsymbol{\omega}^\star$ found to the SMF.

```
$start\_time \leftarrow system\_time$; $\boldsymbol{\omega} \leftarrow N^{SMF}_{RR}$; $\boldsymbol{\omega}^\star \leftarrow \boldsymbol{\omega}$; $no\_local\_impr \leftarrow 0$; $no\_global\_impr \leftarrow 0$;
while $system\_time \leq time\_limit + start\_time$ do
    $h \leftarrow 1$;
    $T_p \leftarrow \emptyset$ for all $p \in \{1, 2, \dots, 6\}$;
    while $no\_global\_impr < max\_no\_global\_impr$ do
        while $no\_local\_impr < max\_no\_local\_impr$ do
            $trial\_solution\_set \leftarrow \{s(\boldsymbol{\omega}) : s \in N^{SMF}_{shift}(\boldsymbol{\omega})\}$;
            $trial \leftarrow tabu$;
            while $trial = tabu$ do
                $\boldsymbol{\omega}' \leftarrow s(\boldsymbol{\omega}) \in trial\_solution\_set$ that has the best $z(s(\boldsymbol{\omega}))$;
                $trial\_solution\_set \leftarrow trial\_solution\_set \setminus \{s(\boldsymbol{\omega})\}$;
                $trial \leftarrow not\_tabu$;
                for $p = 1$ to $6$ do
                    $e_p \leftarrow a_p(s, \boldsymbol{\omega})$;
                    if $e_p \in T_p$ AND $z(\boldsymbol{\omega}') \leq A(e_p)$ then
                        $trial \leftarrow tabu$;
                    end
                    $p \leftarrow p + 1$;
                end
                if $trial \neq tabu$ then
                    $\boldsymbol{\omega}'_h \leftarrow \boldsymbol{\omega}'$;
                    for $p = 1$ to $6$ do
                        $e_{p,h} \leftarrow e_p$;
                    end
                end
            end
            for $p = 1$ to $6$ do
                $T_p \leftarrow T_p \setminus \{e_{p,h-t_p}\}$;
                $T_{q(p)} \leftarrow T_{q(p)} \cup \{e_{p,h}\}$;
                $A(e_{p,h}) \leftarrow$ The best among $\{A(e_{p,h}), z(\boldsymbol{\omega}), z(\boldsymbol{\omega}'_h)\}$;
            end
            if $z(\boldsymbol{\omega}'_h) > z(\boldsymbol{\omega}^\star)$ then
                $\boldsymbol{\omega}^\star \leftarrow \boldsymbol{\omega}'_h$;
                $no\_global\_impr \leftarrow 0$;
            end
            if $z(\boldsymbol{\omega}'_h) > z(\boldsymbol{\omega})$ then
                $no\_local\_impr \leftarrow 0$;
            else
                $no\_local\_impr \leftarrow no\_local\_impr + 1$;
            end
            $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega}'_h$;
            $h \leftarrow h + 1$;
        end
        $\boldsymbol{\omega} \leftarrow N^{SMF}_{ejec}(\boldsymbol{\omega})$;
        if $z(\boldsymbol{\omega}) > z(\boldsymbol{\omega}^\star)$ then
            $\boldsymbol{\omega}^\star \leftarrow \boldsymbol{\omega}$;
            $no\_global\_impr \leftarrow 0$;
        end
        $no\_global\_impr \leftarrow no\_global\_impr + 1$;
        $no\_local\_impr \leftarrow 0$;
    end
    $\boldsymbol{\omega} \leftarrow N^{SMF}_{RR}$;
    if $z(\boldsymbol{\omega}) > z(\boldsymbol{\omega}^\star)$ then
        $\boldsymbol{\omega}^\star \leftarrow \boldsymbol{\omega}$;
    end
    $no\_global\_impr \leftarrow 0$;
end
```

A feasible solution to the SMF is represented by a permutation vector $\boldsymbol{\omega}$ on the set $\{1, 2, \ldots, |I|\}$ whose j-th component $\boldsymbol{\omega}(j) = i$ corresponds to field $i \in I$ being assigned harvesting sequence position $j \in I$. Let t_i denote the time required to harvest field $i \in I$ and define the index set function $U_i(\boldsymbol{\omega})$ to contain the set of fields in $\boldsymbol{\omega}$ that have a sequence value less than that of field i (*i.e.* $U_i(\boldsymbol{\omega}) = \{\boldsymbol{\omega}(u) \in I : u < j,\ \boldsymbol{\omega}(j) = i\}$). Consecutive time instants for which profit values remain approximately constant are amalgamated as a single time period in a bid to reduce the problem instance size. The set of time periods which thus arises is denoted by K. Let the parameter d_k denote the first time instant of period $k \in K$ and let P_{ik} denote the resulting profit from harvesting field $i \in I$ during period k. Then the function

$$\tau_i(\boldsymbol{\omega}) = \sum_{\boldsymbol{\omega}(u) \in U_i(\boldsymbol{\omega})} t_{\boldsymbol{\omega}(u)}, \qquad i \in I \tag{6.20}$$

gives the time instant at which the harvesting of field i begins and the function

$$\theta_i(\boldsymbol{\omega}) = \min_{k \in K} \{k : \tau_i(\boldsymbol{\omega}) - d_k \geq 0\}, \qquad i \in I \tag{6.21}$$

expresses the period during which field i is harvested. Finally, if the indicator function $\chi_{ik}(\boldsymbol{\omega})$ is defined to take the value 1 if $\theta_i(\boldsymbol{\omega}) = k$, or 0 otherwise, the objective function in (6.7) may be expressed as

$$z(\boldsymbol{\omega}) = \sum_{i \in I} \sum_{k \in K} P_{ik} \chi_{ik}(\boldsymbol{\omega}). \tag{6.22}$$

The sequence into which the fields are arranged thus determines the time instant at which each field is harvested, since the field harvesting times are assumed to be independent of the tonnage of cane available. To determine the time period ($k \in K$) during which a field's harvesting time instant occurs, the harvesting times of the preceding fields are summed in (6.20). The sum of those harvesting times is the time that has elapsed before the harvest of the field in question commences. In order to compute the objective function value associated with a particular solution $\boldsymbol{\omega}$, the summation must be performed for every field. Therefore, the procedure required to compute $z(\boldsymbol{\omega})$ requires $O(n^2)$ steps.

If the harvesting times were a function of the tonnage of cane, and not just the field area, the parameter t_i would become a function of $\boldsymbol{\omega}$, thus requiring a reformulation of the SMF. The modelling of the tonnage-dependent harvesting time is omitted here because cutting speed is sufficiently modelled by area [88,126]. Cutting capacity may also be altered by a grower in order to account for varying conditions. Therefore only area is used as the determining factor for the values assigned to the parameter t_i.

The TS adopted to find a good solution ($\boldsymbol{\omega}^\star$) to the SMF comprises several components: a *random restart procedure* (RR), a *shift neighbourhood move* (SN), an *ejection chain compound move* (EC), as well as procedures for handling *tabu restrictions* and *aspiration criteria*, all described below. The TS is mainly guided by two objective function values encountered during the course of its execution; these are the best objective function value in (6.22) encountered so far, denoted by $z^\star$, and the best objective function value encountered since the last EC or RR application, denoted by z'.

The TS begins by applying the SN to the current solution, which returns a new current solution $\boldsymbol{\omega}$. If the objective function value of the new current solution is better than z', an iteration counter denoted by *no_local_impr* is set to 0, otherwise the value of this counter is increased by 1. If the objective function value of the solution is better than $z^\star$, an iteration counter denoted by *no_global_impr* is set to 0. If *no_local_impr* is greater than a certain specified value, the EC is applied to the current solution and the value of an iteration counter *no_global_impr* is increased by 1. If the counter *no_global_impr* is larger than a specified threshold, the RR is applied.

The *random restart procedure* (RR) generates a pseudo-random permutation of a range of integers from 1 to the total number of fields. The RR thus provides a new starting solution for the TS.

The SN finds a best admissible solution in the shift neighbourhood of the current solution. This shift neighbourhood is formed by considering a shift of any field to another position in the harvesting sequence, generating a *trial solution*, denoted by $\boldsymbol{\omega}'$.

Define a *neighbouring pair*, denoted by $e = (i, j)$, where $i, j \in I$ and $i \neq j$, as any two neighbouring fields (in terms of their position in the harvesting sequence) and denote the set of all neighbouring pairs by E. Furthermore, assume that $\boldsymbol{\omega}$ is "closed" by "connecting" its last field with its first field (*i.e.* by wrapping the solution encoding). A sequence $\boldsymbol{\omega}$ of fields may then be partially described by a subset of E, constructed by including those neighbouring pairs which appear in $\boldsymbol{\omega}$. For example, if $\boldsymbol{\omega} = [2, 3, 1]$, the neighbouring pairs are $(2, 3)$, $(3, 1)$ and $(1, 2)$, the rightmost neighbouring pair $(1, 2)$ constituting the closing (or wrapping around) of the sequence.

The move from a current solution to a trial solution may now be described in terms of *deleted* and *added* neighbouring pairs. For example, when moving from $\boldsymbol{\omega} = [2, 3, 1]$ to $\boldsymbol{\omega}' = [3, 2, 1]$ (by shifting field 3 into position 1), none of the three neighbouring pairs $(2, 3)$, $(3, 1)$ or $(1, 2)$ remain neighbouring pairs in $\boldsymbol{\omega}'$, hence these neighbouring pairs are said to have been deleted. The three added neighbouring pairs are $(3, 2)$, $(2, 1)$ and $(1, 3)$.

There are six so-called *tabu lists*, denoted by T_1, T_2, T_3, T_4, T_5 and T_6, respectively, which list neighbouring pairs that either may not be deleted or may not be added. When considering a move from $\boldsymbol{\omega}$ to $\boldsymbol{\omega}'$, the deleted neighbouring pairs are tested for membership in the tabu lists which contain neighbouring pairs that may not be deleted and the corresponding test is also performed for the added neighbouring pairs. An *aspiration function*, denoted by $A(e)$, is defined so as to return a single value as a function of neighbouring pair e. During these membership tests, if any neighbouring pair e is found in any of its corresponding tabu lists, the aspiration function value corresponding to the particular neighbouring pair ($A(e)$) is evaluated according to the *aspiration criterion* $A(e) < z(\boldsymbol{\omega}')$. If the aspiration criterion evaluates as true, the move receives a passing status with respect to the particular neighbouring pair. However, if the aspiration criterion does not evaluate as true, the move is given a non-passing status. The values returned for each neighbouring pair by the function $A(e)$ are updated so as to reflect the best objective function value obtainable by allowing the addition or deletion of neighbouring pair e. The trial solution $\boldsymbol{\omega}'$ with the best objective function value that has achieved passing status for all associated neighbouring pairs is selected to be returned to the TS as the new current solution $\boldsymbol{\omega}$. Then the neighbouring pairs that were added are inserted into the tabu lists that list neighbouring pairs that may not be deleted, while the neighbouring pairs that were deleted are added to the tabu lists that contain neighbouring pairs that may not be added. The tabu lists have specified lengths (tenures) and each time a neighbouring pair is added to any of the lists, the oldest neighbouring pair is removed.

The reason for employing six tabu lists is that it provides for the possibility of enforcing more or less restrictiveness on moves based on the *type* of deleted or added neighbouring pair. If desired, the length of the list that retains the middle added neighbouring pair, for example, could be made longer or shorter as a means of disfavouring or favouring certain kinds of algorithmic behaviour. Alternatively, a passing status may be awarded to a move as long as the associated neighbouring pairs are not present in more than some fraction of the lists.

The EC rearranges the field harvesting sequence, employing rules that facilitate the preservation of parts of the quality of the sequence while seeking to *diversify* the TS. The EC starts by computing a parameter value for each field that measures the relative performance of the field during each period. This is achieved by means of the *performance parameter*

$$\Upsilon_{ij} = P_{ij} \Big/ \max_{k \in K} \{P_{ik}\}, \qquad i \in I, \quad j \in K.$$

Given a particular harvesting sequence $\boldsymbol{\omega}$, each field is assigned a current relative performance as a result of its time of harvest, the time of harvest being computed by means of (6.21). The field with the lowest current relative performance is deemed to have the largest potential for improvement. This field is *ejected* from the sequence, "opening" its position for another field. A sequence with such an "opening" is by convention called a *reference structure* [75, 238]. The ejected field i is reserved until the end of the application of the EC, when it is assigned to the then "open" position. The reference structure has an *improvement potential* g defined as $g = 1 - \Upsilon_{ij}$. All other fields $k \in I,\ k \neq i$ are now compared and the field with the largest value of $\Upsilon_{kj} - \Upsilon_{k\ell}$, is selected to move from position k to position i in the reference structure (corresponding to moving from period ℓ to period j), where ℓ is the current period of the selected field and j is the period associated with the "open" position i. This is called a *reference structure move*. The reference structure move only occurs if the condition $\Upsilon_{kj} - \Upsilon_{k\ell} \geq -g$ holds. If the condition does not hold, the EC stops performing reference structure moves. However, if a reference structure move *does indeed* occur, g takes the value $g + \Upsilon_{kj} - \Upsilon_{k\ell}$ and the EC continues to perform reference structure moves by investigating the remaining fields with respect to their relative performance increase/decrease values, possibly augmenting the chain of reference structure moves with another move. Any field may only be subjected to one reference structure move. The EC ends by assigning the first ejected field to the position of the last ejected field, thus turning the reference structure into a new solution encoding vector $\boldsymbol{\omega}$.

The TS was coded in Microsoft's *Visual Basic for Applications for Excel.* Investigative computational results were obtained on the same machine as for the integer programming instances mentioned in §6.2.1 and are shown in Table 6.3. The results indicate that the solution times necessary to obtain acceptable solutions by means of the TS are stable for instances A–E. Instance F was solved twice (the two runs named F1 and F2, respectively) using different time limits, achieving results approximately 0.01 % apart in terms of optimality. Solutions to practical size instances were obtained on a regular basis during the validation of the DSS presented later in this dissertation, uncovering satisfactory objective function values.

6.3 Chapter summary

Two combinatorial optimisation problems were formulated in this chapter. The BMF formed the core of the DSSDV (which is not described further in this dissertation) and was based on the GAP. It was solved to within an acceptable percentage of optimality by means of a local search algorithm coded in Wolfram's Mathematica.

The SMF constitutes the core of the final version of the tactical harvest scheduling decision support system—the DSS presented later in this dissertation—and was solved using an attribute-based tabu search method incorporating a shift neighbourhood and an ejection chain. Failed attempts to solve small instances of the SMF using LINGO 11 shows that it is a hard problem to solve to optimality. Even its LP relaxation could not be solved for practical size instances

Instance	Problem size (Fields $\times$ Time instants)	CPU time limit [s]	Feasible solution?	Optimality gap [%]
A	5×14	10	Y	0.1100
B	5×28	15	Y	0.2480
C	7×21	30	Y	0.0788
D	8×21	30	Y	0.0100
E	10×21	30	Y	1.0351
F1	60×210	300	Y	≤ 9.55
F2	60×210	900	Y	≤ 9.54

Table 6.3: *SMF instances solved by means of the TS. The time limits are given in seconds. Note that instances D and E did not require an increase in the time limit. The optimality gap was computed as that fraction of the lower bound remaining as a gap between the LP relaxation lower bound and the objective function value of the best solution found. A practical size instance was solved twice (F1 and F2) using a lower bound computed by summing the minimum cost during any period for each field, since no LP-relaxation solution was available.*

(due to Lingo Core Solver memory problems having been encountered when applying Lingo 11 installed on a Dell Optiplex 755, Intel Core2 vPro 3 GHz, 4 Gb RAM workstation). This chapter stands in partial fulfilment of Dissertation Objectives VI and VII of §1.3.

CHAPTER 7

Model parameter estimation

Contents

7.1 Introduction

This chapter contains a description of the regression and other models used to estimate the parameters P_{ij} and M_{ij} introduced in Chapter 6. Given a particular farm or other relatively homogenous area, the current state of each field is taken into consideration during these estimations. Unless otherwise specified, the models presented here are those used in the DSS presented later in this dissertation. The underlying concepts of the methodologies employed are introduced and a rationale behind each estimate is given. As mentioned earlier, the BMF constitutes the mathematical programming formulation employed in the DSSDV, while the SMF is the mathematical programming formulation employed in the final DSS.

The only sets of parameters in the BMF are the field-by-period profits (P_{ij}), the field-by-period yields (M_{ij}), the minimum harvested tonnage per period (D_j^{min}) and the maximum harvested tonnage per period (D_j^{max}). The first parameter represents the value of the sugarcane on a particular field during a particular time period less the cost of harvesting that particular field during that particular time period, expressed in South African Rand. The second parameter represents an estimate of the yield in tonnes of sugarcane on a particular field during a particular time period. The latter two parameter sets are quantities that the user can manipulate until he/she is satisfied with the harvested cane mass across the season and also to ensure that the DRD is taken into consideration. The parameters D_j^{max} or D_j^{min} do not appear in the SMF.

To a grower, the profit from harvesting a field of sugarcane is the remuneration paid by a miller less the costs associated with the harvesting operation and the fixed costs associated with the land allocated to sugarcane on the farm. Since the SMF has been designed to render decision support based on *profit differences* between fields over various time periods, it is logical to ignore those fixed costs that are equal across all fields. Assuming that only the operational costs differ between fields, let

$$P_{ij} = V_{ij} - C_{ij} \tag{7.1}$$

denote the profit associated with harvesting field i during period j, where V_{ij} is the value of the sugarcane on field i during period j and C_{ij} is the cost of recovering that value. Then the value and costs may be treated separately.

7.2 The value component

The value of sugarcane is well-defined in South Africa, once a physical process of sampling and measuring has been put in place. Any cane consignment at a sugar mill is first crushed, then sampled and its three primary constituents [159] are measured: fibre, moisture and brix. *Brix* is the sugar component and consists of sucrose and other sugars. According to the payment system, let S_{ij} be the sucrose content of cane harvested from field i during period j (measured as a percentage of mass), and let d be a coefficient that accounts for molasses value as well as the processing difficulties that non-sucrose causes in the refining process. Furthermore, let N_{ij} denote the non-sucrose content (in fact, molasses contains some sucrose) of cane harvested from field i during period j (measured as a percentage of mass), and let c be a coefficient that accounts for the processing difficulties that arise from fibre itself and its constituents. Finally, let F_{ij} be the fibre content of cane harvested from field i during period j (measured as a percentage of mass). Then the RV %, denoted by R_{ij}, for field i when harvested during period j, may be expressed as

$$R_{ij} = S_{ij} - dN_{ij} - cF_{ij}, \qquad i \in I, \quad j \in J. \tag{7.2}$$

If P is the season's price per tonne of recoverable value, which is re-estimated monthly and published on the South African Canegrowers' website [195], $\bar{y}$ is the seasonal average percentage of recoverable value for the entire mill, $\bar{w}_j$ is the average percentage of recoverable value for the entire mill during period j and M_{ij} is the mass of the cane in field i during period j, then the remuneration (in Rand) for harvesting field i during period j may be expressed as

$$V_{ij} = PM_{ij}(R_{ij} + \bar{y} - \bar{w}_j), \qquad i \in I, \quad j \in J. \tag{7.3}$$

There are several ways of estimating the mass and RV % of sugarcane, such as using multiple linear regression analysis, plant physiology-based weather-driven models and visual appreciation (eye-estimation). The method chosen for the purposes of this dissertation is multiple linear regression. This approach facilitates an estimation of the values of the parameters S_{ij}, N_{ij} and F_{ij} in (7.2) unnecessary, but places a heavy reliance on field records, since they form the data set to which the regression models are fitted.

7.3 Multiple linear regression analysis

Multiple linear regression analysis is a technique within the framework of statistical inference that allows for the fitting of data to a mathematical expression with several variables [23,24,156, 157]. The technique is based on the method of least squares, refined by Gauss during his work on the prediction of the position of the stellar object Ceres during the early 19th century [67]. The expression describes a tentative relationship between a dependent variable and a set of independent variables. The estimated parameters of a multiple linear regression model appear linearly in the model, hence the name "linear regression", but in the mathematical function (the regression model) it may be multiplied by any function of the dependent variables. In fact, the regression model may be composed of any number of terms involving one or several of the independent variables, as long as the estimated parameters are scalars. The data are usually arranged in order of appearance[1] in a column vector denoted by $\boldsymbol{y} = [y_1, y_2, \ldots, y_n]^T$.

The technique rests on the assumptions that any differences (called residuals) between the regression model and the data are distributed according to a normal distribution and are independent of one another. The statistical inferences made during the analysis are not mathematically valid if the residuals are not normally and independently distributed.

Multiple linear regression may be conducted using historical data or data from a designed and controlled experiment. The technique is identical for these two situations; however, care must be taken with conclusions regarding cause-and-effect relationships using historical data. Unknown variables may be responsible for variability attributed to a variable that correlates in some fashion with one or more of the unknown variables.

The technique begins by proposing a model to which to fit the observed data $\boldsymbol{y}$. Let

$$\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{\epsilon},$$

where

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_k \end{bmatrix}, \quad \boldsymbol{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \ldots & x_{1k} \\ 1 & x_{21} & x_{22} & \ldots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \ldots & x_{nk} \end{bmatrix} \text{ and } \boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}.$$

Here β_0 is the model intercept, k is the number of terms in the regression model (excluding β_0), $\beta_1, \beta_2, \ldots, \beta_k$ denote the k regression coefficients, n is the number of observed data values, $x_{i1}, x_{i2}, \ldots, x_{ik}$ denote the values of the *regressor variables* corresponding to data value $i \in \{1, 2, \ldots, n\}$, and $\epsilon_1, \epsilon_2, \ldots, \epsilon_n$ denote the errors between the model and data. The best fit between the data and the model is achieved by treating the parameters $\beta_1, \beta_2, \ldots, \beta_k$ as variables and minimising the sum of the squared values of $\epsilon_1, \epsilon_2, \ldots, \epsilon_k$. This is equivalent to minimising

$$L = \boldsymbol{\epsilon}'\boldsymbol{\epsilon} = (\boldsymbol{y} - \boldsymbol{X\beta})'(\boldsymbol{y} - \boldsymbol{X\beta}) = \boldsymbol{y}'\boldsymbol{y} - \boldsymbol{2\beta}'\boldsymbol{X}'\boldsymbol{y} + \boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{X\beta}.$$

Minimisation is achieved by letting

$$\left.\frac{\delta L}{\delta \boldsymbol{\beta}}\right|_{\hat{\boldsymbol{\beta}}} = \boldsymbol{-2X'y} + \boldsymbol{2X'X\hat{\beta}} = 0$$

[1]In a designed experiment, the data values are ordered in the *standard order* of the experimental design employed. Using historical data, the data are ordered according to an arbitrary but known scheme.

from which the entries of the vector $\hat{\boldsymbol{\beta}}$ may be derived. The result,

$$\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}'\boldsymbol{X}\right)^{-1} \boldsymbol{X}'\boldsymbol{y},$$

makes it simple to compute the regression coefficients from the selected model and the data. The fitted regression model

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}}$$

may subsequently be analysed diagnostically for appropriateness. The diagnostics usually performed on regression models are mainly based on the *residuals*

$$\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}}. \tag{7.4}$$

Diagnostics involve the computation of the R^2-value, the adjusted R^2-value (R^2_{adj}), the *prediction error sum-of-squares* value ($PRESS$) and other key indicators of whether the model may be expected to be a good predictor. The diagnostics also involve—perhaps more importantly—the plotting of residuals against dependent and independent variable values, on normal probability paper[2] and, in the case of a designed experiment, the run order.

The regressor variables are allowed to be functions of other variables, often those variables that represent the true *system factors*[3]. There may, for example, exist a relationship between a regressor variable x_2 and an underlying variable t_2, such that $x_2 = t_2^2$. A regression model including such a term would be classified as polynomial of at least second order. Note, however, that the regression coefficients still appear linearly in the model.

There are several approaches to selecting an appropriate tentative model to fit the data to. The model terms may be selected using past experience, which is appropriate if the phenomenon under study has been rigourously experimented upon and modelled in the past. It may then only be required to estimate the expected value of the parameters of an already well-known relationship. However, some phenomena are more or less unknown, and thus require other model selection approaches. In 1959, Daniel [42] invented the use of the half normal probability[4] plot of effects from factorial experiments to separate effects that are significant. The approach works by calculating all effects (regression model coefficients multiplied by 2) and plotting their absolute values on half-normal probability paper, and the effects that obviously deviate from a thick (pen-width) straight line drawn from $[0, 0]$ through the group of effects that *do* end up on the line, are declared not statistically significant. According to [156], the half-normal probability plot works well with designed and controlled industrial experiments, but is not infallible.

Another good approach towards selecting a regression model is the backward elimination approach. It starts by proposing the greatest order model possible and performs a statistical analysis. The single model term with the lowest statistical significance value on its regression coefficient is eliminated. A new statistical analysis is performed and the least significant term's regression coefficient is again eliminated. The process is continued until no term is statistically insignificant. The procedure may terminate with only the intercept left in the regression model. Miller [152] explored the problem of coefficient selection thoroughly [86]. If the model is to be used for prediction purposes, it must be *hierarchical* [156], which means that any higher order

[2]Normal probability paper has a logarithmically scaled vertical axis designed to cause normally distributed ranked data plot to form a straight line.

[3]The system factors may, for instance, be the row-spacing in metres and fertiliser amount involved when growing sugarcane.

[4]The half normal probability paper has the same vertical axis as normal probability paper, but uses the absolute value of the variable being plotted. The probability papers are equivalent in usage, but look different.

term involving a particular variable requires all possible lower order terms involving that variable to be present in the model. If, for example, $\beta_{12}x_1x_2$ is statistically significant and included in the regression model, then β_1x_1 and β_2x_2 must also be included. During the backward elimination process, terms that are not believed to be sensible to include, *i.e.* terms that suggest relationships that do not make sense in the practical setting being modelled, may be discarded if not significant. Terms that are heavily correlated with each other must be considered with special care, and may sometimes be resolved (by deleting one of them) using previously acquired insight about the process under study. Finally, one very useful tool amidst all statistical tests and indicators normally used in multiple regression analysis, is the simple response surface plot in which both the model output and data are depicted together. Generating such a plot may provide a safeguard against trivial errors and erroneous assumptions. When there are more than two independent variables, multiple plots are required.

There is an abundance of statistical software available for analysis of multivariate regression models. To name a few, IBM SPSS's *SPSS* [198], SAS Institute's *SAS* [180], StatSoft's *Statistica* [205], Wolfram's *Mathematica* [149], MathWorks' *Matlab* [150], Minitab Inc.'s *Minitab* [155] and StatEase's *Design Expert* [204] all have extensive regression modelling capability. Important and useful books on linear regression analysis are [23, 24, 156, 157].

7.4 The yield and RV models

In the DSS presented later in this dissertation, matrices $\boldsymbol{M} = [M_{ij}]$ and $\boldsymbol{R} = [R_{ij}]$ are populated by localised values obtained from farm regression models which are, in turn, fitted to historical data from the farm in question. When such data are not available, data from a farm that operates under similar conditions and in a similar fashion are used instead. The values of $\bar{y}$ and $\bar{w}_j$ are based on weekly data of RV % for all South African sugar mills (the SASRI data set) provided by SASRI [183]. As noted in §2.5, SASRI's CANEGRO model [183] is unable to accommodate RV %. The expression in (7.2), where only the coefficients d and c are known (determined by corporate entities as laid out in the sugar industry agreement [49]), contains the parameter S_{ij} and two further unknowns. CANEGRO is able to estimate S_{ij}, but not the remaining two unknowns, thus making it impossible to estimate R_{ij} using CANEGRO. This finding is supported in the claim by Bezuidenhout *et al.* [18] that CANEGRO cannot simulate RV %.

In (7.3) the seasonal average RV % for the entire mill, $\bar{y}$, and the periodic average percentage of RV %, $\bar{w}_j$, for the mill area during period j are required. The methodology for estimating $\bar{y}$ is to use the Mill Group Board estimate for the season, or to use the average RV % from historical data over a reasonable number of years. The value of $\bar{w}_j$ may be computed by means of simple regression on the SASRI data set.

The case study area selected for this purpose is located in the Eston mill region in KwaZulu-Natal and four growers (who co-own a harvesting syndicate) originally agreed to contribute with data and schedule evaluation during the 2009 season, and later agreed to continue doing so during the 2010 season. One of these farms, Seafield farm, is used as an example throughout this chapter and is situated 10 kilometres north of Richmond.

7.4.1 Base yield models

As mentioned earlier, the method chosen to estimate the yield of a field at a given moment in time in the future is multiple linear regression analysis. The models were, from the onset, expected to be polynomials, but which model terms should be included and the degree of the included model terms were not known. The model selection approach was adapted to the available field records.

For practical modelling purposes, a definition of cane age other than the cane's actual age was introduced. The notion is called *effective growth time* (EGT) and is the age of the cane in months, not counting the time spent during months of slow or halted growth. In the case study area, June, July and August are classified as such "no-growth" months.

In order to fit the regression models, the historical data for a particular farm are stratified according to variety, so that cane varieties are associated with one regression model each. Using Seafield farm as an example, the data spanned the time period 2002 to 2007, and there were sufficient data to model the cane varieties N12, N16, N35 and N37. There were not sufficient data to distinguish between different crop classes, different years, different field aspects, or different field toposequences. Figure 7.1 shows the results of both a first-order model with EGT in months as regressor (x) and yield in t.ha^{-1} for the N12 variety as response (y) and a second-order model involving the same regressor and response. The accompanying residual plots indicate that the second-order model is superior. The R^2-values of the two models are 0.975 for the first-order model and 0.986 for the second-order model. The *base yield model* for N12 is $y = 11.7x - 0.29x^2$.

The intercept in both models should be fixed[5] at zero in order to ensure that—particularly under data scarcity—the regression model always displays a parabolic shape with a single maximum extreme point as well as abiding by the fact that cane with EGT 0 always yields 0 t.ha^{-1}. The fixing of the intercept is the cause of the relatively large R^2-values. The ANOVAs along with the respective model functions for the four varieties are shown in Figure 7.2 and the graphical representations of the base yield models for N16, N35 and N37 are shown in Figure 7.3.

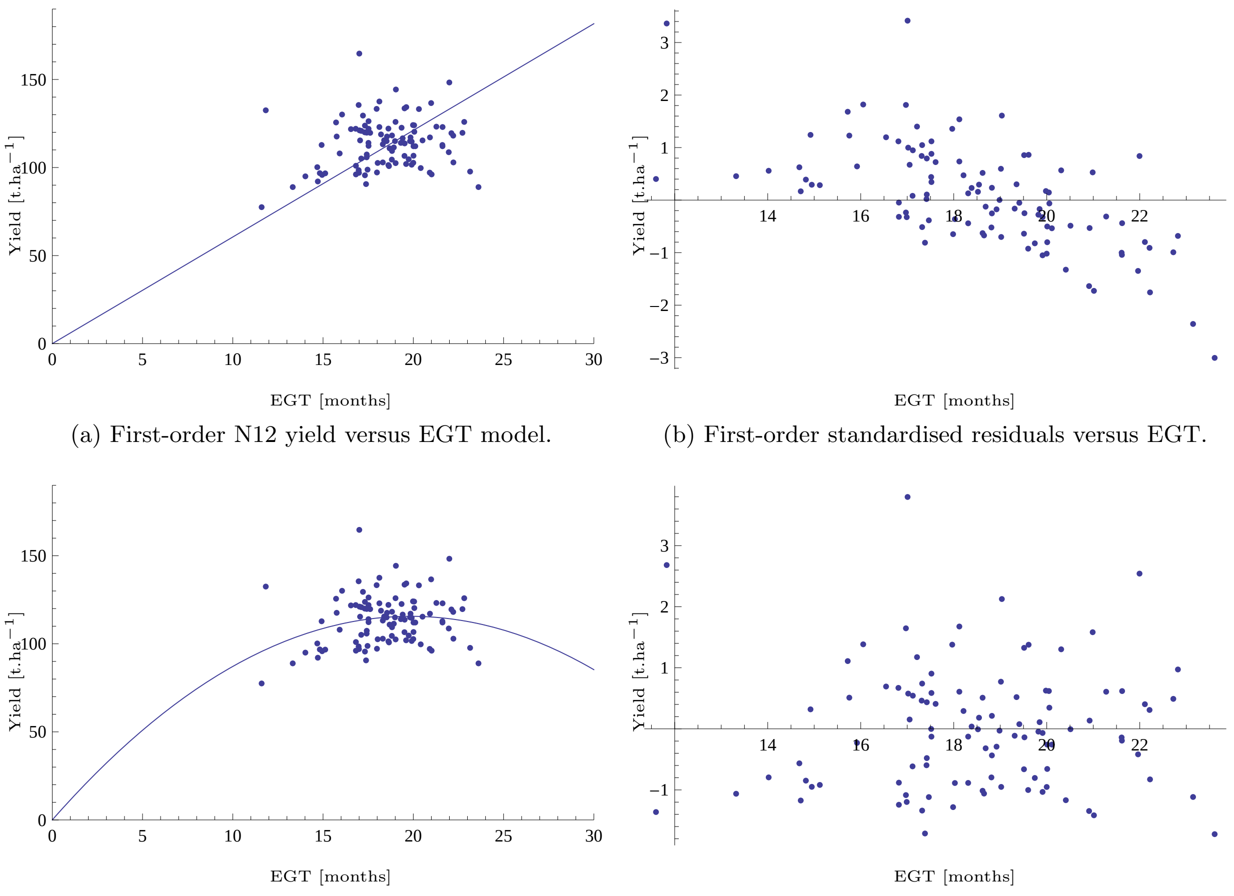
In order to compute a base yield model value B_{ij}^M for field i during period j, the variety present on the field is first established (the information about variety is available in the field records). Secondly, the EGT of the field is computed and the future EGT of the field at the midpoint of period j is also computed. The EGT of field i at the midpoint of period j is denoted by a_{ij}. For example, a field at Seafield farm containing the cane variety N12, with $i = 4$, $j = 5$ and $a_{45} = 21.2$ months has a B_{ij}^M-value of

$$\begin{aligned} B_{45}^M &= \beta_1 a_{45} + \beta_{11} a_{45}^2 \\ &= 11.7 \times 21.2 - 0.29 \times 21.2^2 \\ &= 117.7 \quad \left[\text{t.ha}^{-1}\right]. \end{aligned}$$

7.4.2 Event-driven yield models

The ability to account for factors other than the regressors was found during the case study to be crucial to the usefulness of the resulting DSS. This ability was partly incorporated by means of so-called *event-driven yield models.* Event yield models are simple first-order polynomials where yield is a function of the number of days elapsed since some time marker.

[5]When the intercept is fixed, the computation of the R^2-value is affected.



(a) First-order N12 yield versus EGT model.

(b) First-order standardised residuals versus EGT.

(c) Second-order N12 yield versus EGT model.

(d) Second-order standardised residuals versus EGT.

Figure 7.1: *A comparison of a first-order model and a second-order model of yield per hectare for the cane variety N12 with EGT as the regressor at Seafield farm. The standardised residuals versus EGT plots indicate that the second-order model is more appropriate.*

If an extraneous event $\ell_i(k) \in E^x$ takes place on field i, the time in days between its occurrence and the midpoint of period j is first computed; that time is denoted by $q_{ij\ell_i(k)}$, where $k \in \{1, 2, 3\}$ is the order number of the k^{th} event[6] that has taken place on the particular field. Let E^M_{ij} denote an event-driven yield model value that adjusts yield in t.ha^{-1} during period j to account for events that have occurred on field i, let $\delta^M_{0,ij\ell_i(k)}$ denote the step decrease due to the k^{th} event and let $\delta^M_{1,ij\ell_i(k)}$ denote the rate of yield decrease in t.ha^{-1}d^{-1} (tonnes per hectare per day after the day of the event) due to the k^{th} event. Then E^M_{ij} may be expressed as

$$E^M_{ij} = \sum_{k=1}^{3} \left(\delta^M_{0,\ell_i(k)} + q_{\ell_i(k)} \delta^M_{1,ij\ell_i(k)} \right) \qquad i \in I, \quad j \in J. \tag{7.5}$$

To compute M_{ij}, the base yield model and the event-driven yield model are added together to yield

$$M_{ij} = B^M_{ij} + E^M_{ij}, \qquad i \in I, \quad j \in J. \tag{7.6}$$

[6]Note that only as many as three different events are kept on record, assuming that any serious events will simply replace less serious ones, should a fourth or further extraneous event occur.

N12	Df	SS	MS	p-value
x	1	1313880	1313880	< 0.0001
x^2	1	14743.6	14743.6	< 0.0001
Error	101	18828.7	186.4	
Total	103	1347460		

(a) N12 model $y = 11.7x - 0.29x^2$.

N16	Df	SS	MS	p-value
x	1	267247	267247	< 0.0001
x^2	1	6195.5	6195.5	< 0.0001
Error	24	5369.7	223.7	
Total	26	278812		

(b) N16 model $y = 11.1x - 0.28x^2$.

N35	Df	SS	MS	p-value
x	1	113981	113981	< 0.0001
x^2	1	1944.7	1944.7	0.03
Error	13	4538.2	349.1	
Total	15	120464		

(c) N35 model $y = 12.0x - 0.39x^2$.

N37	Df	SS	MS	p-value
x	1	107603	107603	< 0.0001
x^2	1	515.1	515.1	0.003
Error	5	91.5	18.3	
Total	7	108209		

(d) N37 model $y = 14.0x - 0.38x^2$.

Figure 7.2: *Four yield models fitted to data from Seafield farm. Here Df denotes degrees of freedom, SS denotes sum-of-squares and MS denotes mean square.*

For realism, negative values of M_{ij} may be rounded up to 0, but there is a drawback to such an approach. If many of the fields suffer from severe events concurrently, there may not be enough time to harvest them all before their yields have reached zero, at which stage some of the fields will receive no further increase in postponement penalty (because it is being rounded up to zero). If a suffering field is not scheduled for harvesting within the time before the yield reaches zero, it may be scheduled arbitrarily anywhere in the remainder of the harvesting sequence. The same is the case for rounding RV %. On the other hand, if the field is worthless already, it may or may not cause any harm to leave it to be harvested later during the season.

There are a total of twenty-three types of events in the set E^x, out of which thirteen were included to affect yield. As mentioned, $\delta^M_{0,\ell_i(k)}$ and $\delta^M_{1,ij\ell_i(k)}$ are the respective coefficients that determine the step change and rate of change of yield. The area fraction H_{ik} ($i \in I$, $k \in \{1,2,3\}$) affected by the event and the event's interactions with variety, field toposequence and field aspect ($V^E_{i\ell_i(k)}$, $L^E_{i\ell_i(k)}$ and $F^E_{i\ell_i(k)}$) affect the values of the coefficients which may be expressed as

$$\delta^M_{0,ij\ell_i(k)} = d^M_{0,\ell_i(k)} H_{ik} V^E_{i\ell_i(k)} L^E_{i\ell_i(k)} F^E_{i\ell_i(k)}, \qquad i \in I, \quad j \in J, \quad k \in \{1,2,3\}, \quad \ell_i(k) \in E^x, \quad (7.7)$$

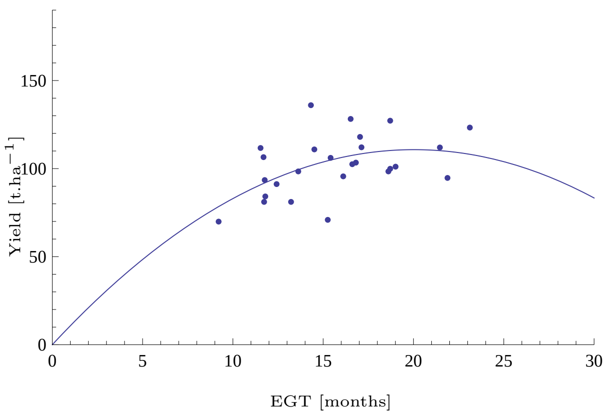
$$\delta^M_{1,ij\ell_i(k)} = d^M_{1,\ell_i(k)} H_{ik} V^E_{i\ell_i(k)} L^E_{i\ell_i(k)} F^E_{i\ell_i(k)}, \qquad i \in I, \quad j \in J, \quad k \in \{1,2,3\}, \quad \ell_i(k) \in E^x, \quad (7.8)$$

where the *yield event coefficient* $d^M_{0,\ell_i(k)}$ is the best estimate available of the mean step change in t.ha^{-1} for the geographical region in which the farm is located and the yield event coefficient $d^M_{1,\ell_i(k)}$ is the best available estimate of the mean change in yield per day, measured in t.ha^{-1}d^{-1}.

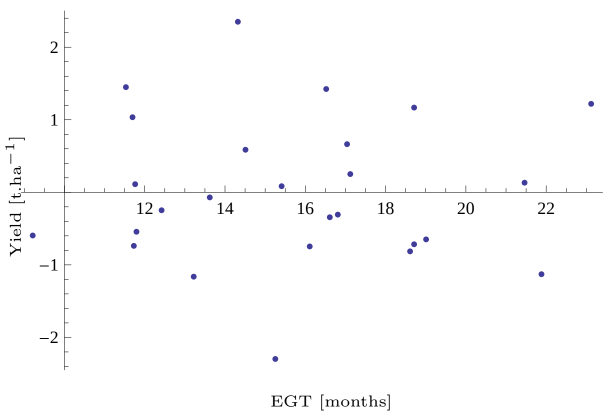
7.4.3 Base RV models

As for the base yield models, one *base RV model* per farm and variety is fitted. The regressor is Julian date[7]. As for the yield data, the RV % data are stratified according to variety, so that there is one base RV % model per farm and variety. The data from Seafield farm were sufficient for fitting regression models in the case of sugarcane varieties N12, N16 and N37, and a second-order polynomial again provided the best fit. Contrary to the base yield models, the intercept in the base RV models was not set to zero (since RV % is not necessarily zero on

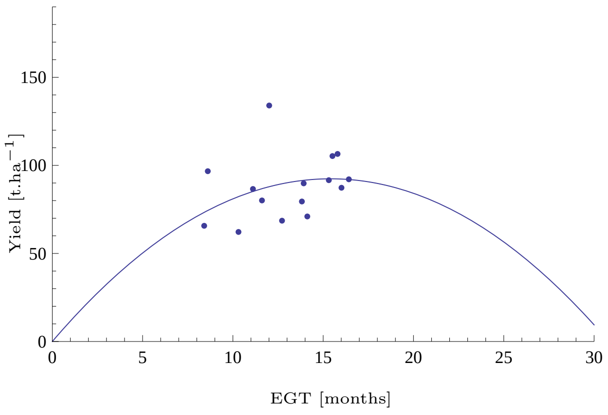
[7]The Julian date is the number of the day counting from January 1st.



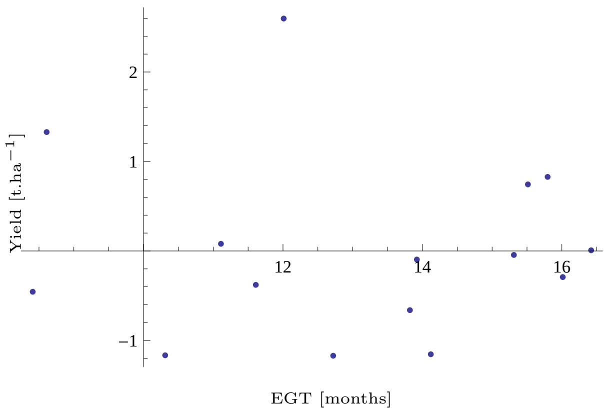
(a) N16 base yield model.



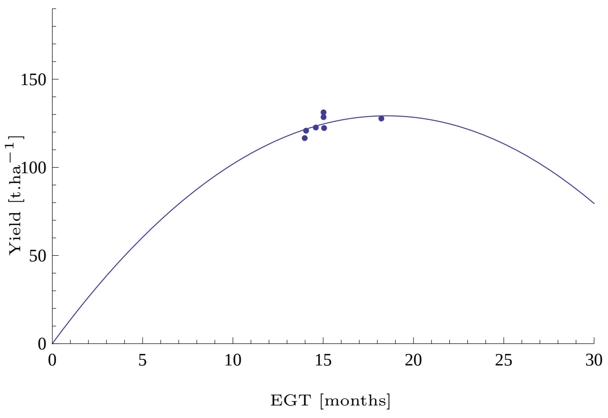
(b) N16 standardised residuals *vs.* EGT.



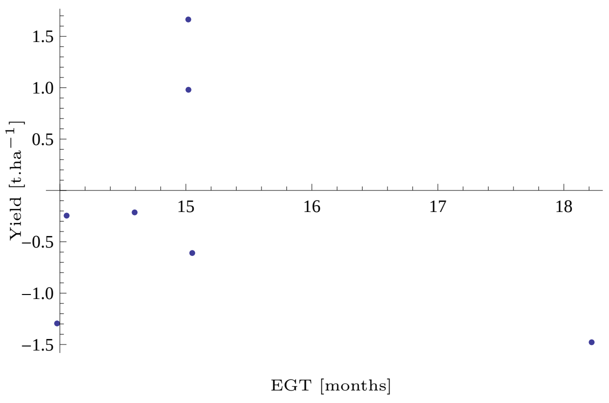
(c) N35 base yield model.



(d) N35 standardised residuals *vs.* EGT.



(e) N37 base yield model.



(f) N37 standardised residuals *vs.* EGT.

Figure 7.3: *Three base yield models for cane varieties N16, N35 and N37, along with residual plots. Notice the scarcity of data for the N37 base yield model.*

January 1st). Figure 7.5 shows the results of the three ANOVAs as well as the base RV models. The base RV models are denoted by B^R_{ij}.

7.4.4 Event-driven RV models

In order to account for the effect on RV % caused by the occurrence of extraneous events, *event-driven RV models* are introduced, similar to the usage of event-driven yield models described in §7.4.2. The notation for the event-driven RV models follows that of the event-driven yield models, where $q_{ij\ell_i(k)}$ denotes the time in days between successive occurrences of event $\ell_i(k) \in E^x$ on field i and the midpoint of period j. The order number $k \in \{1,2,3\}$ retains the same meaning as previously. Let E^R_{ij} denote an event-driven RV model value that adjusts RV % in unit percent during period j to account for events that have occurred on field i, let $\delta^R_{0,ij\ell_i(k)}$ denote the step decrease due to the kth event and let $\delta^R_{1,ij\ell_i(k)}$ denote the RV % decrease in unit % per day after the day of the event due to the kth event. Then E^R_{ij} may be expressed as

$$E^R_{ij} = \sum_{k=1}^{3} \left(\delta^R_{0,\ell_i(k)} + q_{\ell_i(k)} \delta^R_{1,ij\ell_i(k)} \right), \qquad i \in I, \quad j \in J. \tag{7.9}$$

To compute R_{ij}, the base RV model and the event-driven RV model are summed together to yield

$$R_{ij} = B^R_{ij} + E^R_{ij}, \qquad i \in I, \quad j \in J. \tag{7.10}$$

Twelve events were included to affect RV %. The field area as well as the event's interactions with variety, field toposequence and field aspect affect the values of the RV event coefficients in the same fashion (and to the same degree) as for yield, so that

$$\delta^R_{0,ij\ell_i(k)} = d^R_{0,\ell_i(k)} H_{ik} V^E_{i\ell_i(k)} L^E_{i\ell_i(k)} F^E_{i\ell_i(k)}, \qquad i \in I, \quad j \in J, \quad k \in \{1,2,3\}, \quad \ell_i(k) \in E^x, \tag{7.11}$$

$$\delta^R_{1,ij\ell_i(k)} = d^R_{1,\ell_i(k)} H_{ik} V^E_{i\ell_i(k)} L^E_{i\ell_i(k)} F^E_{i\ell_i(k)}, \qquad i \in I, \quad j \in J, \quad k \in \{1,2,3\}, \quad \ell_i(k) \in E^x, \tag{7.12}$$

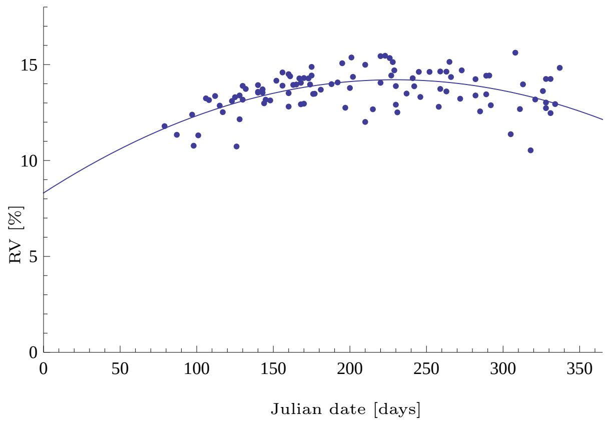
where the *RV event coefficient* $d^R_{0,\ell_i(k)}$ is the best estimate available of the mean step change RV % in unit % due to the kth event for the geographical region in which the farm is located and the RV event coefficient $d^R_{1,\ell_i(k)}$ is the estimate of the mean change in RV % in unit % per day due to the same event.

7.5 The cost component

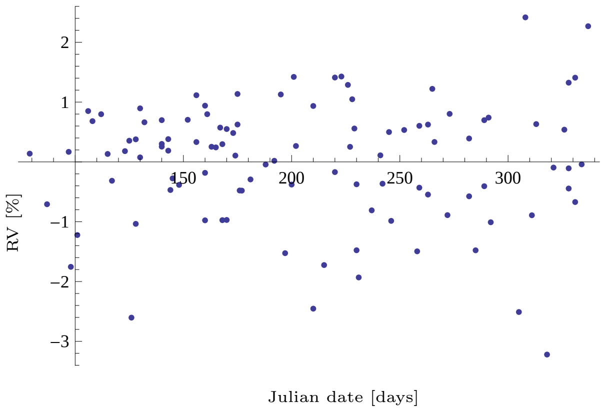
The scheduling model within the DSS takes various costs into consideration, such as the cost of cutting the sugarcane, loading, reloading, transporting it to the mill and other operational costs. The scheduling model is sometimes forced to behave in a certain manner, which is achieved by means of penalties in the form of large costs.

7.5.1 Event-driven costs

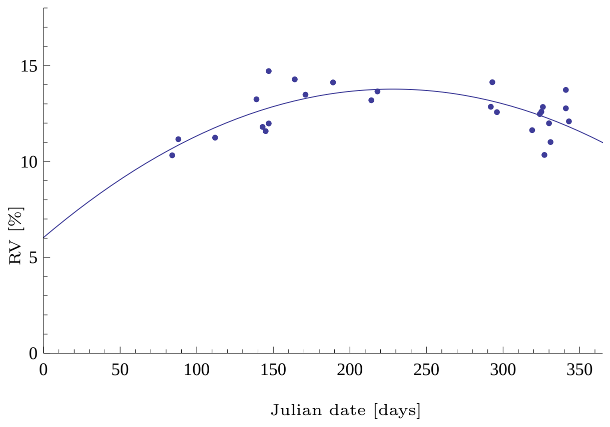
The DSS also imposes penalties in the form of costs as a means of discouraging solutions to the SMF that are deemed beforehand to be undesirable. For example, if a grower marks a field



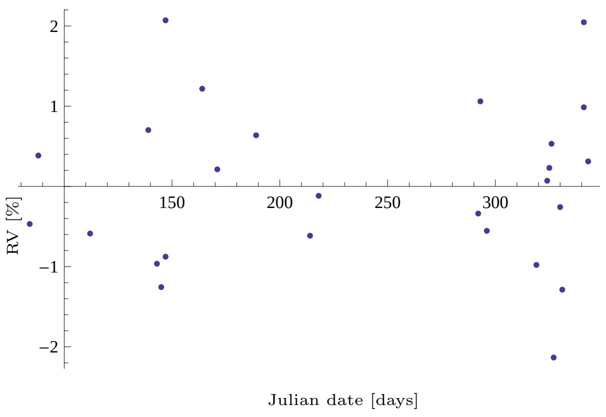
(a) N12 RV versus Julian date model.



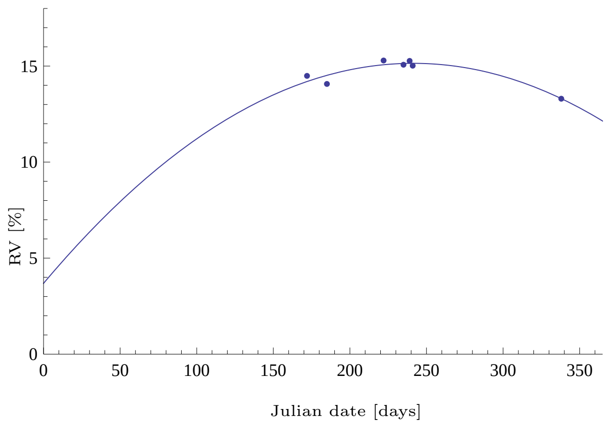
(b) N12 standardised residuals *vs.* Julian date.



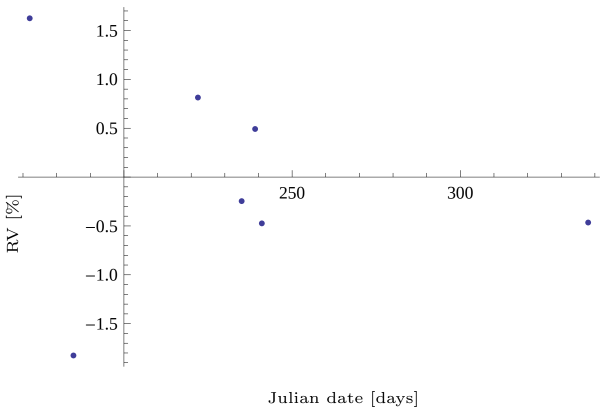
(c) N16 RV versus Julian date model.



(d) N16 standardised residuals *vs.* Julian date.



(e) N37 RV versus Julian date model.



(f) N37 standardised residuals *vs.* Julian date.

Figure 7.4: *Three RV models fitted to data from Seafield farm.*

N12	Df	SS	MS	p-value
x	1	5.06	5.06	0.013
x^2	1	28.2	28.2	< 0.0001
Error	100	79.0	0.79	
Total	102	112		

(a) N12 model $y = 8.31 + 0.051x - 0.00011x^2$.

N16	Df	SS	MS	p-value
x	1	0.063	0.063	0.79
x^2	1	14.7	14.7	0.00061
Error	23	21.5	0.94	
Total	25	36.3		

(b) N16 model $y = 6.04 + 0.068x - 0.00015x^2$.

N37	Df	SS	MS	p-value
x	1	0.62	0.62	0.061
x^2	1	2.30	2.30	0.0076
Error	4	0.37	0.093	
Total	6	3.29		

(c) N37 model $y = 3.69 + 0.095x - 0.00020x^2$.

Figure 7.5: *Three RV % models fitted to data from Seafield farm. Here Df denotes degrees of freedom, SS denotes sum-of-squares and MS denotes mean square.*

with an *early plough-out decision*[8], a penalty is applied to the field if scheduled for harvesting after the 181st day of the year (end of June). This approach has led to the introduction of the notions of a *harvesting time window* (HTW) and a *harvesting time deadline* (HTD) which specify upper bounds on the number of days to wait before harvesting some affected field and the date before which to harvest some affected field, respectively. For example, a field affected by an Eldana infestation receives an HTW of 20, and the cost matrix C_{ij} is manipulated by adding a penalty $c_{ij}^{HTW} = P^E q_{ij\ell_i(k)}$, where P^E is a per-day-per-tonne penalty coefficient. When C_{ij} is computed, the Julian date of the event is subtracted for each $j \in J$ from the first Julian date of the period. The portion of that difference that exceeds 20 is multiplied by the penalty coefficient and subsequently added to the computation of C_{ij}. A realistic estimate for the value of P^E is 10 Rand per day per tonne [164], and reflects on the importance of adhering to the time windows and deadline dates.

7.5.2 Fixed operational costs

The operational costs involved when harvesting and transporting sugarcane arise under the completion of a number of activities, which, for a typical harvesting and transport operation in South Africa involves at least:

- oversight of the harvesting operation,
- when cutting manually: burning the fields before cutting,
- when cutting mechanically: burning is optional,
- transporting the cutters and drivers to, from and between fields,
- transporting or driving mechanised equipment between fields,
- cutting the cane,

[8]The early plough-out event is a decision—not an environmental event—taken early in the season by the grower to plough out and replant a field during the latter parts of the season. Common farming practice is to harvest such a field early in order to optimise total yield.

	$V^E_{i\ell_i(k)}$								$L^E_{i\ell_i(k)}$			$F^E_{i\ell_i(k)}$							
Event	N12	N16	N23	N29	N35	N37	N40	N41	H	M	V	NW	N	NE	E	SE	S	SW	W
Drought death	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Drought stress	.9	1.2	1	1.2	1.4	1.1	1.1	.9	1.2	1	.8	1.1	1.2	1.1	1	.9	.8	.9	1
Eldana borer	.9	1.2	1	.9	1.2	1	1	.9	1.1	1	.9	1.1	1.2	1.1	1	.9	.8	.9	1
Cool fire	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Warm fire	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Flowering	1	.8	1.2	1.2	.8	.8	1	.8	1	1	1	1	1	1	1	1	1	1	1
Frost -2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Frost -3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Frost -5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Frost -7	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Frost -9	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Frost black	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Green leaf sucker	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Lodging	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Mosaic	1	1	1	1	.8	.8	.8	1	1	1	1	1	1	1	1	1	1	1	1
Normal	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Plough out early	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Red rot	.8	1	.8	.8	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Ripening 6 week	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Ripening 8 week	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
RSD	1.1	1.1	1.1	1.2	1	1	.9	1.2	1	1	1	1	1	1	1	1	1	1	1
Sesamia borer	1	1	1	1	1	1	1	1	1.1	1	.9	1.1	1.2	1.1	1	.9	.8	.9	1
Wetness	-	-	-	-	-	-	-	-	.7	1	1.3	.9	.8	.9	1	1.1	1.2	1.1	1

Table 7.1: *The interaction coefficients $V^E_{i\ell_i(k)}$, $L^E_{i\ell_i(k)}$ and $F^E_{i\ell_i(k)}$. The values of $V^E_{i\ell_i(k)}$ were chosen as numerical representations of varietal properties as described in SASRIs Information Sheets [182]. The values of $L^E_{i\ell_i(k)}$ and $F^E_{i\ell_i(k)}$ were conservatively assigned based on discussions with the owner of Seafield farm [164] and the case study harvesting operation manager [126].*

- topping the cane,
- windrowing, stacking or bundling the cane,
- driving long-range transport vehicles and tractors to and into the fields from other fields, a mill or their depots,
- loading the cane from the ground into bins, trailers or long-range transport vehicles using mechanised equipment,
- transporting the cane from the field to the mill, waiting and unloading it.

It is complicated to estimate the exact cost of these harvesting operations, which are typically an unknown function of the day of the year and the geographical location of the field. However, in KwaZulu-Natal, there are estimates of manual harvesting costs at 30 Rand per tonne of cane with infield loading and 46 Rand per tonne with the use of an intermediary loading zone [126, 164]. The cost of transportation is heavily dependent on the distance between the field and the mill, but for the case study area mentioned earlier it is assumed to be approximately 30 Rand per tonne [164]. These costs include pay-off of capital, fuel, wages and other direct costs such as contractors. This may seem a gross over-simplification, but most fields belonging to the same farm differ only marginally in terms of their distances to the mill. The fields on a particular farm in South Africa that lie furthest apart from one another may be at a distance of, for example, 50 km and 55 km from the mill, respectively. The harvesting operation only travels directly between fields if they are harvested on the same day. Otherwise the harvesting operation returns to a *depot* every afternoon, bringing personnel and equipment along. Harvesting different fields on the same day becomes more common the larger the farm or estate. The notation for the cost of harvesting and the cost of transportation for field i during period j is c_{ij}.

7.5.3 Rain-dependent operational costs

It is not so important to supply the DSS with perfectly calculated costs per tonne of cane as it is to estimate differences in costs due to inherent differences between fields or the occurrence of extraordinary events. One such difference occurs due to the practice of *zone loading*, mentioned in Chapter 3, which entails the additional loading and unloading of tractor-trailer combinations. An approach towards modelling this practice is described in this section.

The fields may be classified according to differences in the manner in which they may be accessed during harvesting operations. Fields are said to belong to one of two *access categories*: *access category 1* (AC1) or *access category 2* (AC2). Fields in AC1 are accessible to a long-range transport vehicle when dry and a tractor when wet. Fields in AC1 are level, have a layout conducive to in-field loading and have good access roads. Fields in AC1 are zone loaded only if they are wet. Fields in AC2 are only accessible to tractors, when dry or wet. Roads connecting AC2 fields may be poor or steep, or the fields themselves may be on slopes too steep for large, long-range vehicles to operate on. Fields in AC2 are zone loaded at all times.

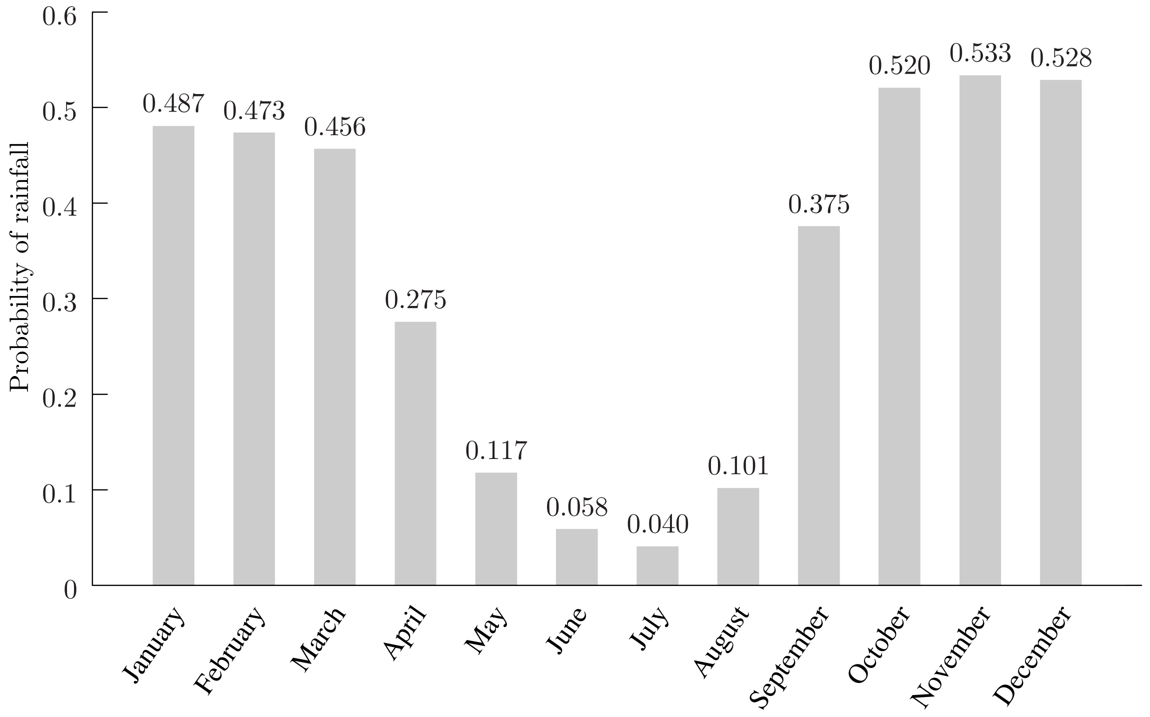


Figure 7.6: *The fraction of days that received at least one instance of precipitation by month, averaged over the years 2000 to 2007.*

No field, regardless of category, may be harvested during precipitation. The surface of the field weakens and softens when it is wet. Hence driving any type of vehicle into the field during rainy weather causes damage to the cane stools and surface layer of the soil. The differences in costs between the above-mentioned categories are rather assumed to consist of the necessary zone loading and the capacity decrease that occurs when conditions are wet, which is the case

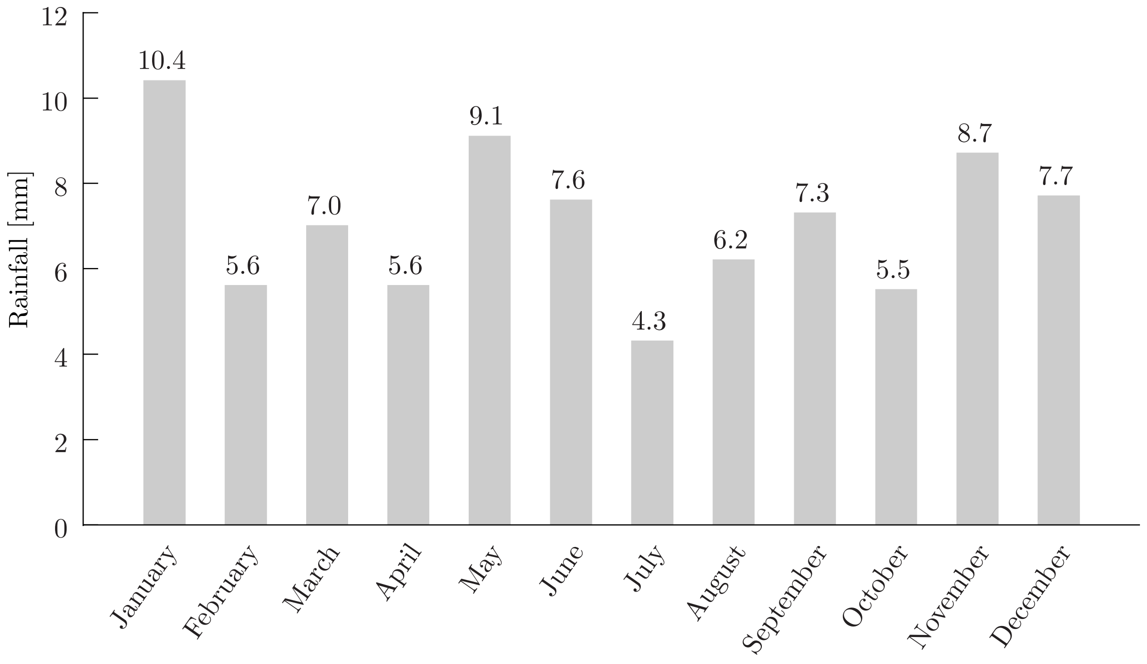


Figure 7.7: *The average amount of precipitation in millimetres received on the days that received precipitation by month, averaged over the years 2000 to 2007. The grand average is 7.1 millimetres per day.*

during a period of time after precipitation. Wet conditions cause other problems as well, such as reducing the effectiveness of the burning practice [126] or increasing the amount of soil being picked up and mixed with the cane during loading.

Figure 7.6 shows that during the winter months of June, July and August, precipitation only occurs on approximately one out of twenty days. The average amount of precipitation on days that actually experience precipitation is shown in Figure 7.7 and the greater the amount of precipitation, the longer the field will take to dry out.

During the part of the season that receives relatively few instances of precipitation, the fields usually dry out within a few hours, making them almost always accessible to long-range transport vehicles, tractors and loaders. For the case study area, fields in AC1 may almost always be loaded onto long-range transport vehicles in-field during May, June, July and August, but not as often during April and September, and rarely during October, November, December, January and February.

The approach towards modelling this situation is based on associating a cost with the harvesting of a field of a particular access category during a particular period of the season. Let W_{ij} be a random variable that takes the value 1 if conditions for field i are wet during period j, or 0 otherwise. Fields are assumed to receive equal amounts of precipitation but are not assumed to dry out at equal rates. Accordingly, let $P\left(W_{ij}=1\right)$ denote the probability of wet conditions at field i during period j.

Assuming that the probability of wet conditions is constant during any particular period, the scenario may be modelled by means of an $M/M/1$ queueing system. Let $1/\mu_{ij}$ be an exponentially distributed random variable that denotes the average time for an initially dry field i to dry out after a single occurrence of precipitation (one rain shower) during period j. A unit of precipitation is assumed to be of a certain average millimetre amount for each period,

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
λ_j^{poh}	0.02	0.019	0.011	0.005	0.002	0.002	0.004	0.016	0.022	0.022	0.022	0.022
ψ_j^{apa}	10.4	5.6	7.0	5.6	9.1	7.6	4.3	6.2	7.3	5.5	8.7	7.7
μ_j^{poh}	0.03	0.06	0.05	0.06	0.04	0.05	0.08	0.06	0.05	0.06	0.04	0.05

Table 7.2: *Seasonal values of the number of precipitation occurrences per hour (λ_j^{poh}), the average amount of rainfall during each occurrence (ψ_j^{apa}) expressed in millimetres and the average dry-out rate expressed in number of precipitation occurrences per hour (μ_j^{poh}).*

the amount following an exponential distribution. Furthermore, let $1/\lambda_j$ be an exponentially distributed random variable denoting the average time between precipitation occurrences at all fields during period j. Finally, if $\rho_{ij} = \lambda_j/\mu_{ij}$, the probability of wet conditions may be expressed as

$$\begin{aligned} P(W_{ij} = 1) &= 1 - P(W_{ij} = 0) \\ &= 1 - (1 - \rho_{ij}) \\ &= \rho_{ij}, \qquad i \in I, \quad j \in J, \end{aligned} \tag{7.13}$$

(see, for example, [80]).

For the purposes of estimating μ_{ij} and subsequently ρ_{ij}, the field's aspect and toposequence must be known. The average dry-out time is assumed to depend on the average amount of precipitation per precipitation occurrence, the field aspect (north-west, north, north-east, east, south-east, south, south-west or west) and the field toposequence (hilltop, mid-slope or valley bottom). A field situated mid-slope is assigned the nominal dry-out rate, while a field situated on a hill-top is assumed to dry out 30 % faster and a field situated in a valley bottom is assumed to dry out 30 % slower. A field facing east or west is given a nominal dry-out rate, a north-east or north-west facing field is assumed to dry out 10 % faster while a south-east or south-west facing field is assumed to dry out 10 % slower. A field facing north is assumed to dry out 20 % faster than the nominal rate, while a south-facing field is assumed to dry out 20 % slower than the nominal rate.

The wet probability for a particular field during a particular time period is found through the following computational steps: First, for each period j of the season, an average number of precipitation occurrences per hour λ_j^{poh} is computed using the information in Figure 7.6. For January, for instance, the computation becomes $0.48/24 = 0.020$ occurrences per hour. An assumption is that the days that receive precipitation only receive a single occurrence; this assumption was made to enable the next step. Secondly, an average precipitation amount of water in millimetres per occurrence ψ_j^{apa} is extracted from Figure 7.7, which for January is 10.4 millimetres per occurrence. Thirdly, it is assumed that the average dry-out rate μ^{apa} is 0.35 millimetres per hour, but that the dry-out rate μ_j^{poh} in number of precipitation occurrences per hour is expressed as $\mu_j^{poh} = \mu^{apa}/\psi_j^{apa}$, which for the January example becomes $\mu_j^{poh} = 0.35/10.4 = 0.034$ occurrences per hour. The field-specific dry-out rate in number of occurrences per hour is then expressed as $\mu_{ij} = \mu_j^{poh}/(L^E_{i\ell_i(k)}F^E_{i\ell_i(k)})$, where $\ell_i(k)$ represents the event called "Wetness" in Table 7.1. Using (7.13), the probability of wet conditions becomes $P(W_{ij} = 1) = \rho_{ij} = \lambda_j^{poh}/\mu_{ij}$. In the above computation the end result may occasionally be a value larger than 1; this, however, is not of concern, since its only use is to multiply it by a penalty coefficient. The resulting values of λ_j^{poh}, ψ_j^{apa} and μ_j^{poh} for the case study area are presented in Table 7.2.

Let ρ_{ij} from (7.13) be the risk of wet conditions on field i on a day during period j and let P_i^W be the penalty, *i.e.* an estimate of the cost to the business, incurred if field i is scheduled to be harvested on a wet day. Then the *zone loading cost* component, c_{ij}^d, becomes

$$c_{ij}^d = P_i^W \rho_{ij} \qquad i \in I, \quad j \in J. \tag{7.14}$$

The penalty P_i^W depends only on the accessibility of field i, which for the fields in AC 1 is equal to the cost of zone loading (estimated at 16 Rands per tonne [126] and for the fields in AC 2 is equal to zero (AC 2 fields are always zone loaded, so their c_{ij}^d-values are instead fixed at 16 Rand per tonne). In summary, the cost component C_{ij} may be expressed as

$$C_{ij} = c_{ij} + c_{ij}^{HTW} + c_{ij}^{HTD} + c_{ij}^d, \qquad i \in I, \quad j \in J. \tag{7.15}$$

7.6 The profit

The various coefficients necessary to complete the computations described in §7.4 and §7.5 are summarised in Table 7.3. Given that M_{ij}, R_{ij}, C_{ij}, $\bar{y}$, $\bar{w}_j$ and the RV-price P have been computed according to the above procedures, the profit P_{ij} associated with harvesting field i during period j should be computed by determining the value of V_{ij} using (7.3) and substituting this value into (7.1).

7.7 Chapter summary

The various models employed to associate a value and a cost with the harvesting of a field during some time period were introduced and described in detail in this chapter. The value of harvesting a field is estimated by means of the sugarcane value, which has been modelled by means of a regression analysis of the yield, a regression analysis of the recoverable value, special models of the effects of extraneous events on yield and recoverable value as well as the price of recoverable value and the regional conditions involved in sugarcane production. The cost of harvesting the field is modelled by estimating the cost of the activities involved, separated into two parts: costs under dry conditions and costs under wet conditions. All other aspects of the costs involved in the harvesting and transportation process are aggregated as far as possible. The appraisal of the value and cost of harvesting a particular field during a particular time period takes (adverse) extraneous events into consideration in a novel fashion. This chapter is thus in partial fulfilment of Dissertation Objective VI of §1.3.

Event	$d^M_{1,l(k)}$	Yield effect assumptions	$d^R_{0,l(k)}$	$d^R_{1,l(k)}$	RV % effect assumptions	HTW	HTD
Drought death	−0.383	Deterioration computed to cancel out 1/4 of the maximum N12 growth rate.	0	−2	RV % estimated to deteriorate by 14 unit % per week.	365	365
Drought stress	−0.383	See drought death.	0	0		365	365
Eldana borer	0		0	−0.0015	0.045 unit % RV % deterioration per month.	20	365
Cool fire	−0.038	Deterioration estimated at one tenth of the maximum N12 growth rate.	0	0		365	365
Warm fire	−0.383	See drought death.	0	−0.45	Old (>18 months) crops deteriorate by 0.45 unit % RV % per day.	365	365
Flowering	−0.109	Deterioration estimated at 10 $t.ha^{-1}$ per 3 months.	0	0		365	365
Frost −2	0	Present for record keeping purposes.	0	0		365	365
Frost −3	−0.038	See cool fire.	0	0		365	365
Frost −5	−0.383	See drought death.	0	0		21	365
Frost −7	−0.383	See drought death.	0	−1	1 unit % per day.	14	365
Frost −9	−0.383	See drought death.	0	−2	2 unit % per day.	7	365
Frost black	−0.383	See drought death.	0	−2	2 unit % per day.	7	365
Green leaf sucker	−0.038	See cool fire.	0	−0.0007	Half of the Eldana depreciation.	365	365
Lodging	−0.109	See flowering.	0	−0.0108	0.33 unit % per month.	365	365
Mosaic	0	Should be harvested early if severe.	0	0		365	181
Normal	0	Indicates that no events have occurred.	0	0		365	365
Plough out early	0	Indicates that grower has decided to plough out field. Should be harvested early.	0	0		365	181
Red rot	0		0	−0.0007	See Green leaf sucker.	365	365
Ripening 6 week	0		2	0	Increases RV % by 2 unit %. Should be harvested within 42 days.	42	365
Ripening 8 week	0		2	0	Increases RV % by 2 unit %. Should be harvested within 56 days.	56	365
RSD	−0.153	Slows growth by 40 % of the N12 maximum growth rate.	0	0	Should be harvested early.	365	181
Sesamia borer	0		0	−0.0007	See Green leaf sucker.	365	365

Table 7.3: *Yield and RV event coefficients as well as* harvesting time windows *(HTW) and* harvesting time deadlines *(HTD) for the twenty-three events incorporated into the DSS presented later in this dissertation. HTW is the number of days counting from the day of the event within which the field should be harvested, while HTD is the Julian date before which the field should be harvested. The reason for not displaying $d^M_{0,l(k)}$ is that it is 0 for all the currently incorporated events.*

CHAPTER 8

The decision support system

Contents

This chapter contains descriptions of a conceptual architecture of a DSS put forward to aid with sugarcane harvest scheduling decisions. This architecture contains various building blocks which are also described as well as the workflow and dataflow related to these building blocks. Finally, a computer implementation of the DSS is described in some detail.

8.1 The decision support system as a concept

From a high level, the DSS may be viewed as a tool which operates in a stand-alone manner on a personal computer, which needs input in the form of relatively readily available historical data on field level productivity and which recommends a harvesting schedule for a single, current season, as shown graphically in Figure 8.1.

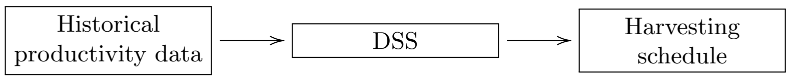


Figure 8.1: *High-level view of the DSS inputs and outputs.*

8.1.1 The DSS architecture

The DSS architecture is designed for the *medium-scale solitary commercial grower* and *medium-scale commercial harvesting group* contexts. The building blocks that make up the DSS architecture are called the *user interface building block*, the *run data building block*, the *field database building block*, the *coefficient database building block*, the *prediction models building block* and the *scheduling model building block*. The architecture is shown schematically in Figure 8.2.

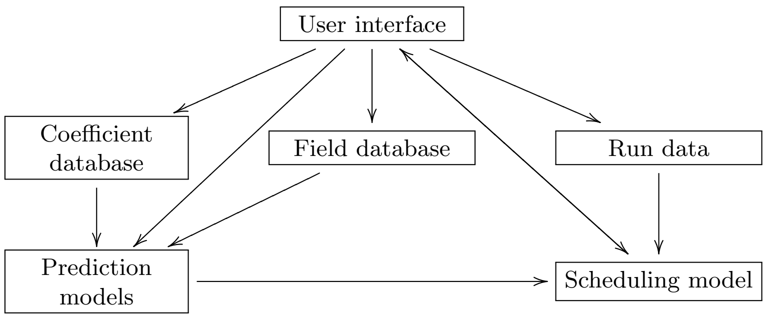


Figure 8.2: *Conceptual DSS architecture.*

The user interface provides the means by which to populate the coefficient database building block, to populate the field database building block, to enter information into the run data building block, and to control the performance of the prediction models and scheduling model as well as methods for producing legible harvesting schedules. The coefficient database building block stores all coefficients for the prediction models of Chapter 7, while the field database building block stores all field data required for the correct application of the prediction models building block. The run data building block contains information required for the correct application of the scheduling model building block.

The prediction models building block contains the models presented in Chapter 7, namely the cane yield models, the recoverable value percentage prediction models, the cane yield event models, the recoverable value percentage event models, the event penalisation models, the harvesting cost models, the wet conditions probability models, the relative recoverable value payment system models, the mill area seasonal average prediction model and the mill area periodical average prediction model.

The scheduling model building block houses the model and solution methods presented in §6.2.2, namely the sequential model formulation and the attribute based tabu search with a shift neighbourhood and an ejection chain.

8.1.2 The building blocks

Several of the conceptual building blocks mentioned above were indirectly described in Chapters 6 and 7. In this section, the building blocks are further described within the context of the DSS architecture.

The *user interface building block* provides the user with the possibility of entering values into the *run data building block*, the *coefficient database building block*, the *field database building block* as well as controlling the performance of the *prediction models building block*, the running of the *scheduling model building block* and the parameter values within the embedded tabu search algorithm.

The *run data building block* consists of values for harvesting and transportation costs, zone-loading costs, season starting and ending dates, event-related penalty coefficients P^E, parameter settings for the scheduling model and the RV price. The parameters employed in the TS are *max_no_local_impr*, *max_no_global_impr*, the tabu list tenures of the six tabu lists, and a time limit for the scheduling algorithm.

The *coefficient database building block* contains

- the regression coefficients β_0, β_1 and β_{11} for the base yield models of §7.4.1 and the base RV models of §7.4.3,
- the coefficients for event interactions with variety, field toposequence and field aspect $V^E_{i\ell_i(k)}$, $L^E_{i\ell_i(k)}$ and $F^E_{i\ell_i(k)}$,
- the yield event coefficients $d^M_{0,\ell_i(k)}$ and $d^M_{1,\ell_i(k)}$ as well as the RV event coefficients $d^R_{0,\ell_i(k)}$ and $d^R_{1,\ell_i(k)}$ of §7.4.2 and §7.4.4,
- the zone loading penalty P^W_i, dry-out rate μ^{apa}, the precipitation amount per occurrence ψ^{apa}_j, and the precipitation occurrence rate λ^{poh}_j of §7.5.3.

The *field database building block* contains

- the set of all fields, I,
- the field-by-field values of variety, the previous harvest date, the harvesting front, the aspect, the field toposequence, the field area, the field accessibility, the estimated yield, and the manual yield adjustment,
- the events $\ell_i(k)$ and percentage of crop affected H_{ik}.

The *prediction models building block* contains

- functions for computing the field EGT and the day of the year for future periods $j \in J$,
- the base yield models B^M_{ij} of §7.4.1,
- the base RV models B^R_{ij} of §7.4.3,
- the event-driven yield models E^M_{ij} of §7.4.2,
- the event-driven RV models E^R_{ij} of §7.4.4, including intermediary computations of $\delta^M_{0,ij\ell_i(k)}$, $\delta^M_{1,ij\ell_i(k)}$, $\delta^R_{0,ij\ell_i(k)}$ and $\delta^R_{1,ij\ell_i(k)}$ due to event $\ell_i(k)$,
- a function for computing the zone loading component c^d_{ij} including intermediary computations of λ_j and μ_{ij} of §7.5.3,
- functions for computing the event related waiting and time window penalties c^{HTW}_{ij} and c^{HTD}_{ij} of §7.5.1,
- functions for computing M_{ij}, R_{ij}, $\bar{y}$, $\bar{w}_j$, and finally

- functions for computing V_{ij}, C_{ij} and P_{ij}.

The *scheduling model building block* contains the model and solution approach of §6.2.2. The input taken from the run data building block is the time limit, tabu list tenures, the maximum number of non-improving iterations between ejection chains, the maximum number of non-improving iterations between newly generated randomised starting solutions, and the season starting and ending dates. The data taken from the prediction models building block are the P_{ij}, while the field areas are taken from the field database building block (used to approximate the required time to harvest each field).

A possible approach towards deploying this conceptual architecture in practice is now briefly described. Figure 8.3 shows the various steps required to deploy the DSS together with a reference to what kind of expertise is required by the deployer or which role-player may be suitable to perform each step of the deployment process. The role-players are divided into the three categories indicative of the type of decision each is expected to make on a regular basis, namely the *strategist*, the *manager* and the *consultant*. The strategist is considered to be a person usually responsible for strategic decisions concerning the harvesting operation, such as variety selection, layout planning, and other decisions with long-term or large financial implications. The manager is considered a person normally in charge of daily operations, labour management, short-term planning and who generally has some set of directions to follow. The consultant is someone who is capable of managing the deployment of the DSS and who understands the nature and workings of the models included in the DSS.

The DSS may only be deployed within a delimited harvesting area, which may constitute a single farm or multiple farms. Within this area, the consultant is tasked to populate a finite number of prediction models with parameter values, for example by means of a model fitting technique such as the Method of Least Squares which renders values for the parameters of the model. One way of limiting the number of models is to, for example, decide to fit one model per cane variety and agroclimatically homogenous area, or one model per cane variety and farm. The regressors to include are assumed to depend on the quality and quantity of the available field record data. The strategist then divides the harvesting operation into harvesting fronts and selects fields to be harvested during the current season, assigning each field to a harvesting front. Each harvesting front is treated as a separate problem by the DSS. The manager then gathers data on each field and populates the field database, while the consultant populates the coefficient database and selects solver parameters taking the number of fields into consideration. Finally, a time limit for the solver is specified and the solver is called to produce a harvesting schedule. Once the deployment has been completed, the manager may iterate through the manager level of the diagram in Figure 8.3 each time a new schedule is sought.

8.2 The DSS implemented on a personal computer

This section contains a description of the appearance and functionality of a computer implementation of the DSS described in §8.1. This computer implementation was performed with the main focus of providing a platform from which to test and validate the concepts within the architecture described in §8.1.1 as well as the architecture as a concept in itself.

The components of the DSS implementation are described by means of screen-shots with accompanying descriptions of the functionality of the various sub-components. The implementation is a macro-activated Microsoft Excel workbook and each of its parts is described according to which worksheet in the workbook it is located in. The worksheets are first described with respect

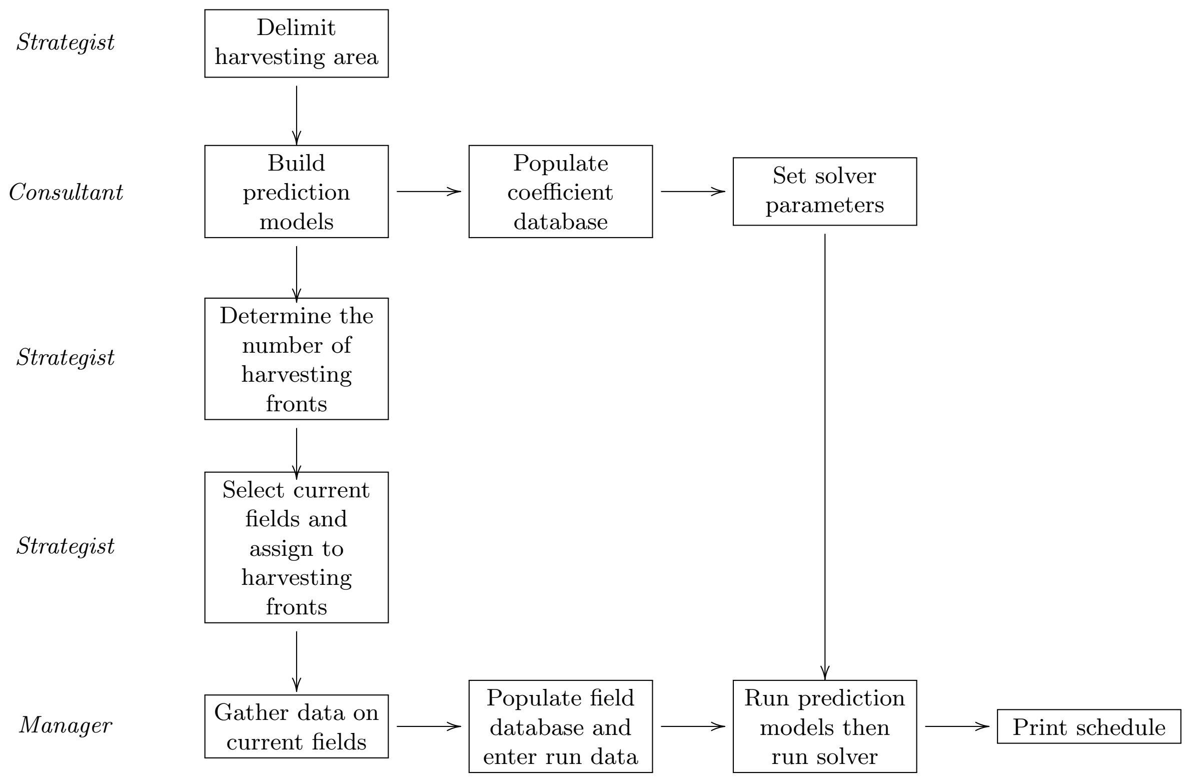

Strategist
Delimit harvesting area
Consultant
Build prediction models
Populate coefficient database
Set solver parameters
Strategist
Determine the number of harvesting fronts
Strategist
Select current fields and assign to harvesting fronts
Manager
Gather data on current fields
Populate field database and enter run data
Run prediction models then run solver
Print schedule

Figure 8.3: *Workflow involved when deploying the DSS described generically in §8.1.*

to their relevance to the user, and are subsequently comprehensively described with regards to functionality. The underlying Visual Basic for Applications for Excel (VBA) source code is included in Appendix A. For the purposes of the implementation development and verification work, two of the worksheets include a search-report feature and a random farm generator.

8.2.1 The user interface and the run data building blocks

The *user interface* mainly consists of the two worksheets “Start” and “HarvestPlan.”

The worksheet “Start”

The most important parts of the user interface are located on the worksheet named “Start” of which a screen-shot is shown in Figures 8.4 and 8.5. The components of Figures 8.4 and 8.5 have been divided into six numbered parts which are described according to their respective user-relevant components. In the following description, there are some values (numeric and otherwise) which the user may specify. In this description, such input is consistently put inside quotation marks, but only the value inside the quotation marks should be typed into the cells in the DSS.

Part 1 of the sheet in Figure 8.4 contains that aspect of the user interface which connects with the field database, where the user may enter information on up to 999 fields. For each field (row) there are 21 columns into which the user may enter values and one column (**DET**) whose values are returned by the DSS, and these 22 columns are now described by heading:

HF contains the numbers or names of the harvesting fronts to which each field belongs. The user may use a numerical value or text to distinguish as many harvesting fronts as desired. The scheduling algorithm only schedules fields within a single selected harvesting front.

FN contains the names of the fields, which need only be unique within each harvesting front. These names may be specified in text or number form.

FA contains the field sizes in hectares.

S contains the toposequences of the fields, where an "H" represents a field lying on a hill-top, where "M" represents a field lying on a mid-slope, and where "V" represents a field lying in a valley-bottom.

FD contains the aspects of the fields, where the strings "N", "S", "E", "W", "NW", "NE", "SE" and "SW" may be input to indicate the direction of the field's normal.

AC contains the access categories of the fields. An access category value of "1" should be specified when a long-range transport vehicle may enter the field whenever it is dry[1], while a "2" should be specified if long-range transport vehicles may never enter the field.

V contains the names of the cane varieties present on the fields. The available names are stored on the worksheet "V", but may, for example, follow the cane variety names used in Chapter 2 ("N12"–"N50"). This variable value is important since the prediction models building block relies solely on this value when selecting the regression coefficients for modelling the cane yield or RV % for each field. If this column is populated carefully, it may accommodate a single model for each field. This is achieved by choosing values for each field that are unique and using them as input in this column, then fitting a cane yield prediction model (base yield model) and an RV % prediction model (base RV model) for each field, not just for each cane variety.

CC contains the crop classes of the fields, where "0" means a plant-crop, "1" means a first ratoon and so on.

LHD contains the last harvest dates of the fields. The last harvest date of a field is the date on which the field was last harvested. This value must be specified in a date format such as "2007/04/24" and should reflect the date from which the age of the cane is to be calculated. The age and EGT are calculated automatically by the DSS.

ESTD contains the user estimate date. The user estimate date is a date usually reported to the local Mill Group Board for which a grower prepares cane yield estimates for all fields on his/her farm. Extensive effort is invested by the grower in estimating the cane yield at this particular date, which is usually taken to be the same date for every field. For example, the grower may choose June 20 as the user estimate date, in which case he should input"2009/06/20" as ESTD.

ESTY contains the estimated field cane yields at ESTD. As indicated above, the grower estimates (or sometimes eye-estimates) the cane yield at the user estimate date for the fields and this estimate is specified in the ESTY-column.

ESTRV contains the estimated field RV percentages at ESTD.

[1] If the field belongs to category 1, the cost of loading its sugarcane is less than that of a category 2 field, but the cost depends on the risk of wet conditions. For more information, see Chapter 7.

EV1 contains the name of event 1 which may have occurred on the various fields. The available names are listed in the worksheet "EV," and should be input exactly as they appear. Optionally, the user may right-click on the cell and select the option "Select from list...," in which case a drop-down list containing all event names will appear, from which the user can select one event.

EV1D contains the dates of the start of or occurrence of event 1 for the various fields. The format is "yyyy/mm/dd" and the value has implications on yield, RV % and cost trends.

EV1 % contains the percentages of the areas affected by event 1 for the various fields, and has implications on yield, RV % and cost trends.

EV2 and EV3 contain additional events that have occurred on the field. The maximum number of events documented for any field is three.

DET contains the differences between the user's estimates and the DSS's estimates at ESTD, and is thus not for user input.

MAV contains manual adjustment values specified by the user when it is considered necessary to increase or decrease the yield value produced by the DSS. The values specified in this column are added as constants to the appropriate regression functions within the DSS and thus affects the yield trends.

Part 2 of the sheet in Figure 8.4 contains the necessary input for defining a current season:

Start of year is specified as a date ("yyyy/mm/dd") and should normally be set as the first day of the current calendar year.

Start of season is specified as a date ("yyyy/mm/dd") and should express the first day of the current milling season.

End of season is specified as a date ("yyyy/mm/dd") and should represent the last day of the current milling season.

RV-price is the current RV-price expressed in Rands per tonne of RV.

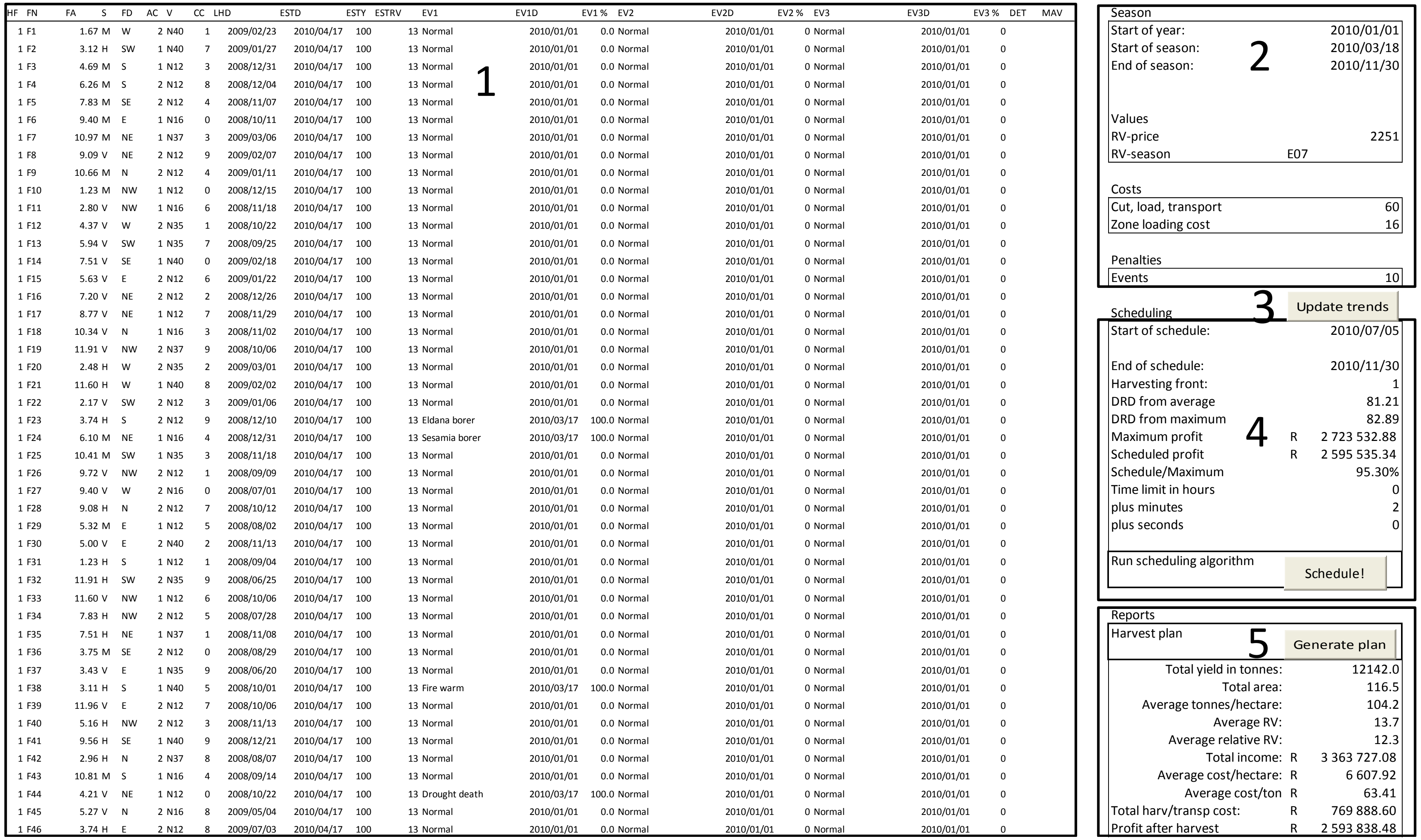
RV-season is one of the models of historical mill area-wide RV % data, specified by entering the name of the chosen season. The available seasons may be found on the worksheet "V", and are represented by regression functions.

Cut, load, transport contains the user-specified harvesting cost as a Rands-per-tonne figure.

Zone loading cost contains the extra cost incurred should an access category 1 field be wet[2].

Events contains a Rands-per-tonne-per-day figure specified by the user for use by the DSS to penalise harvesting postponement of fields suffering as a result of particular events either past a certain event duration or past a certain date. For example, a field with an "Eldana" event is penalised after 20 days after EVD1 and a "Plough out early"-field is penalised after 30 June.

[2]This figure is multiplied by the risk of wet conditions to yield a penalty for scheduling such a field during periods with heightened risk of wet conditions.



HF	FN	FA	S	FD	AC	V	CC	LHD	ESTD	ESTY	ESTRV	EV1	EV1D	EV1 %	EV2	EV2D	EV2 %	EV3	EV3D	EV3 %	DET	MAV
1	F1	1.67	M	W	2	N40	1	2009/02/23	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F2	3.12	H	SW	1	N40	7	2009/01/27	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F3	4.69	M	S	1	N12	3	2008/12/31	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F4	6.26	M	S	2	N12	8	2008/12/04	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F5	7.83	M	SE	2	N12	4	2008/11/07	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F6	9.40	M	E	1	N16	0	2008/10/11	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F7	10.97	M	NE	1	N37	3	2009/03/06	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F8	9.09	V	NE	2	N12	9	2009/02/07	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F9	10.66	M	N	2	N12	4	2009/01/11	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F10	1.23	M	NW	1	N12	0	2008/12/15	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F11	2.80	V	NW	1	N16	6	2008/11/18	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F12	4.37	V	W	2	N35	1	2008/10/22	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F13	5.94	V	SW	1	N35	7	2008/09/25	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F14	7.51	V	SE	1	N40	0	2009/02/18	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F15	5.63	V	E	2	N12	6	2009/01/22	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F16	7.20	V	NE	2	N12	2	2008/12/26	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F17	8.77	V	NE	1	N12	7	2008/11/29	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F18	10.34	V	N	1	N16	3	2008/11/02	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F19	11.91	V	NW	2	N37	9	2008/10/06	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F20	2.48	H	W	2	N35	2	2009/03/01	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F21	11.60	H	W	1	N40	8	2009/02/02	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F22	2.17	V	SW	2	N12	3	2009/01/06	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F23	3.74	H	S	2	N12	9	2008/12/10	2010/04/17	100	13	Eldana borer	2010/03/17	100.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F24	6.10	M	NE	1	N16	4	2008/12/31	2010/04/17	100	13	Sesamia borer	2010/03/17	100.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F25	10.41	M	SW	1	N35	3	2008/11/18	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F26	9.72	V	NW	2	N12	1	2008/09/09	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F27	9.40	V	W	2	N16	0	2008/07/01	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F28	9.08	H	N	2	N12	7	2008/10/12	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F29	5.32	M	E	1	N12	5	2008/08/02	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F30	5.00	V	E	2	N40	2	2008/11/13	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F31	1.23	H	S	1	N12	1	2008/09/04	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F32	11.91	H	SW	2	N35	9	2008/06/25	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F33	11.60	V	NW	1	N12	6	2008/10/06	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F34	7.83	H	NW	2	N12	5	2008/07/28	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F35	7.51	H	NE	1	N37	1	2008/11/08	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F36	3.75	M	SE	2	N12	0	2008/08/29	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F37	3.43	V	E	1	N35	9	2008/06/20	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F38	3.11	H	S	1	N40	5	2008/10/01	2010/04/17	100	13	Fire warm	2010/03/17	100.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F39	11.96	V	E	2	N12	7	2008/10/06	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F40	5.16	H	NW	2	N12	3	2008/11/13	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F41	9.56	H	SE	1	N40	9	2008/12/21	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F42	2.96	H	N	2	N37	8	2008/08/07	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F43	10.81	M	S	1	N16	4	2008/09/14	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F44	4.21	V	NE	1	N12	0	2008/10/22	2010/04/17	100	13	Drought death	2010/03/17	100.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F45	5.27	V	N	2	N16	8	2009/05/04	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		
1	F46	3.74	H	E	2	N12	8	2009/07/03	2010/04/17	100	13	Normal	2010/01/01	0.0	Normal	2010/01/01	0	Normal	2010/01/01	0		

Season

Start of year:		2010/01/01
Start of season:		2010/03/18
End of season:		2010/11/30

Values

RV-price		2251
RV-season	E07	

Costs

Cut, load, transport	60
Zone loading cost	16

Penalties

Events	10

Scheduling

Start of schedule:		2010/07/05
End of schedule:		2010/11/30
Harvesting front:		1
DRD from average		81.21
DRD from maximum		82.89
Maximum profit	R	2 723 532.88
Scheduled profit	R	2 595 535.34
Schedule/Maximum		95.30%
Time limit in hours		0
plus minutes		2
plus seconds		0

Run scheduling algorithm

Reports

Harvest plan

Total yield in tonnes:		12142.0
Total area:		116.5
Average tonnes/hectare:		104.2
Average RV:		13.7
Average relative RV:		12.3
Total income:	R	3 363 727.08
Average cost/hectare:	R	6 607.92
Average cost/ton	R	63.41
Total harv/transp cost:	R	769 888.60
Profit after harvest	R	2 593 838.48

Figure 8.4: *User interface screen-shot of the DSS implementation described in §8.2. The large numbers and black frames have been added to the screen-shot for annotation purposes.*

Part 3 of the worksheet in Figure 8.4 contains the click-button "Update trends" which triggers the calculation of a cane yield value, an RV % value, a cost value, a mill area RV % value and a relative RV % value for each field for every seventh day of the current year, starting with the period January 1–7 and ending with the period December 24–31, and writes these values into the "FDB"-worksheet. For a large field database, this operation may take several minutes, but it must be performed in order for the scheduling algorithm to receive the correct input.

Part 4 of the user interface in Figure 8.4 contains the necessary user input for specifying what part of the worksheet "FDB" should be used as input by the scheduling algorithm. The worksheet "FDB" contains all fields in the database with cane yields, relative RV values and harvesting costs for all periods, but each scheduling run may only be for a certain subset of those fields and a certain subset of those periods. The user specifies ten values in this part:

Start of schedule contains the first day (a date in the format "yyyy/mm/dd") of the schedule to be generated.

End of schedule contains the date of the last day of the schedule to be generated.

Harvesting front contains the name of the current harvesting front. The fields in the database with this name in the "HF" column will be included in the scheduling, unless they have been harvested to 100 %.

DRD from average is the DRD calculated by the DSS based on the assumption that each field yields its across-the-season average value.

DRD from maximum is the DRD based on the assumption that each field yields the maximum value for the season.

Maximum profit contains an estimate of the maximum possible profit from the set of fields in the schedule assuming that each field is harvested during its optimal period. This value is returned by the DSS after having run the scheduling algorithm.

Scheduled profit is the profit value of the solution returned by the DSS subsequent to executing the scheduling algorithm.

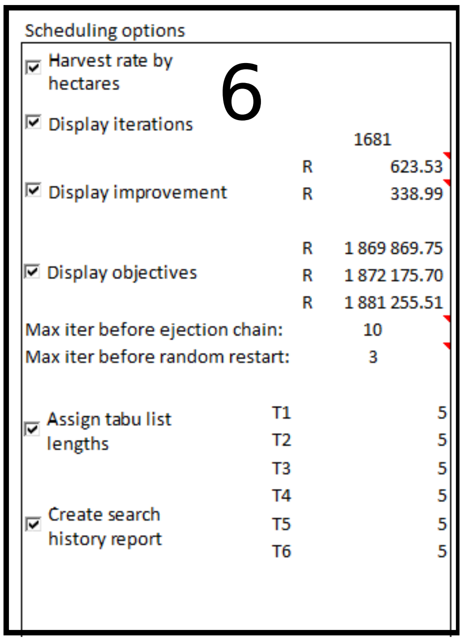
Schedule/Maximum is the percentage of the maximum profit possible that the returned scheduling solution achieved.

Time limit contains the user-specified time limit allocated to the scheduling algorithm (three separate cells; one for the number of hours, one for the number of minutes and one for the number of seconds).

Schedule! is a click-button used to initiate execution of the scheduling algorithm for the current season and harvesting front.

Part 5 of the worksheet in Figure 8.4 contains the click-button "Generate plan" which generates a printable report based on the last solution generated by the scheduling algorithm. A selection of key figures output by the DSS may be found below the click-button and in the worksheet "HarvestPlan" the user may print the schedule as a list excerpt from the database, arranged in order of harvest.

Part 6 of the worksheet is shown in Figure 8.5 and contains run options that are mainly used by the scheduling algorithm, and were included in the DSS implementation for development and debugging purposes.



Scheduling options

✓ Harvest rate by hectares		6
✓ Display iterations		1681
✓ Display improvement	R	623.53
	R	338.99
✓ Display objectives	R	1 869 869.75
	R	1 872 175.70
	R	1 881 255.51
Max iter before ejection chain:		10
Max iter before random restart:		3
✓ Assign tabu list lengths	T1	5
	T2	5
	T3	5
	T4	5
✓ Create search history report	T5	5
	T6	5

Figure 8.5: *User interface screen-shot of the run options part of the user interface of the DSS implementation presented in §8.2. This area of the user interface provides the possibility of manually specifying tabu list tenure and various options for causing the scheduling algorithm to output certain values while running.*

The worksheet "HarvestPlan"

The worksheet "HarvestPlan" contains a list of the fields included in the most recent run of the scheduling algorithm. A screen-shot of the worksheet may be found in Figure 8.6, where the columns contain the following information:

Field contains the names of the fields.

Area contains the areas of the fields (in hectares).

Vrty contains the names of the cane varieties present on the fields.

Month contains the scheduled months of harvest for the various fields.

AgeBrn contains the ages of the fields in months at the scheduled times of harvest.

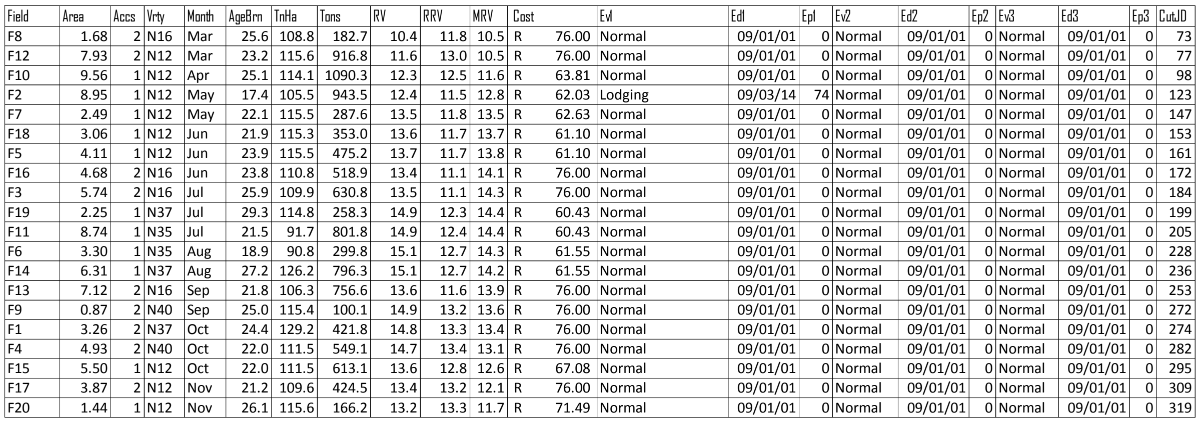
TnHa contains the DSS estimates of the cane yields of the fields (in tonnes per hectare) at the scheduled times of harvest.

Tons contains the DSS estimates of cane yields in tonnes at the scheduled times of harvest.

RV contains the DSS estimate of RV percentages for the various fields at the scheduled times of harvest.

RRV contains the DSS estimates of relative RV percentages at the scheduled times of harvest.

MRV contains the DSS estimates of mill area average RV percentages at the scheduled times of harvest.



Field	Area	Accs	Vrty	Month	AgeBrn	TnHa	Tons	RV	RRV	MRV	Cost	Ev1	Ed1	Ep1	Ev2	Ed2	Ep2	Ev3	Ed3	Ep3	CutJD
F8	1.68	2	N16	Mar	25.6	108.8	182.7	10.4	11.8	10.5	R 76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	73
F12	7.93	2	N12	Mar	23.2	115.6	916.8	11.6	13.0	10.5	R 76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	77
F10	9.56	1	N12	Apr	25.1	114.1	1090.3	12.3	12.5	11.6	R 63.81	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	98
F2	8.95	1	N12	May	17.4	105.5	943.5	12.4	11.5	12.8	R 62.03	Lodging	09/03/14	74	Normal	09/01/01	0	Normal	09/01/01	0	123
F7	2.49	1	N12	May	22.1	115.5	287.6	13.5	11.8	13.5	R 62.63	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	147
F18	3.06	1	N12	Jun	21.9	115.3	353.0	13.6	11.7	13.7	R 61.10	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	153
F5	4.11	1	N12	Jun	23.9	115.5	475.2	13.7	11.7	13.8	R 61.10	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	161
F16	4.68	2	N16	Jun	23.8	110.8	518.9	13.4	11.1	14.1	R 76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	172
F3	5.74	2	N16	Jul	25.9	109.9	630.8	13.5	11.1	14.3	R 76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	184
F19	2.25	1	N37	Jul	29.3	114.8	258.3	14.9	12.3	14.4	R 60.43	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	199
F11	8.74	1	N35	Jul	21.5	91.7	801.8	14.9	12.4	14.4	R 60.43	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	205
F6	3.30	1	N35	Aug	18.9	90.8	299.8	15.1	12.7	14.3	R 61.55	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	228
F14	6.31	1	N37	Aug	27.2	126.2	796.3	15.1	12.7	14.2	R 61.55	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	236
F13	7.12	2	N16	Sep	21.8	106.3	756.6	13.6	11.6	13.9	R 76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	253
F9	0.87	2	N40	Sep	25.0	115.4	100.1	14.9	13.2	13.6	R 76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	272
F1	3.26	2	N37	Oct	24.4	129.2	421.8	14.8	13.3	13.4	R 76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	274
F4	4.93	2	N40	Oct	22.0	111.5	549.1	14.7	13.4	13.1	R 76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	282
F15	5.50	1	N12	Oct	22.0	111.5	613.1	13.6	12.8	12.6	R 67.08	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	295
F17	3.87	2	N12	Nov	21.2	109.6	424.5	13.4	13.2	12.1	R 76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	309
F20	1.44	1	N12	Nov	26.1	115.6	166.2	13.2	13.3	11.7	R 71.49	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	319

Figure 8.6: *A screen-shot showing the HarvestPlan worksheet.*

Cost contains the DSS estimates of the costs—including provisional artificial penalties—in Rands per tonne at the scheduled times of harvest.

Ev1 contains the names of Event 1 for the various fields.

Ed1 contains the dates on which Event 1 occurred or began for the various fields.

Ep1 contains the percentages of the areas that are affected by an Event 1 for the various fields.

Ev2 contains the names of Event 2 for the various fields.

Ed2 contains the dates on which Event 2 occurred or began for the various fields.

Ep2 contains the percentages of the areas that are affected by an Event 2 for the various fields.

Ev3 contains the names of Event 3 for the various fields.

Ed3 contains the dates on which Event 3 occurred or began for the various fields.

Ep3 contains the percentages of the areas that are affected by an Event 3 for the various fields.

CutJD contains the Julian dates of the scheduled harvesting dates for the various fields. This figure is a more precisely expressed variant of the value that column "Month" is based on.

8.2.2 The prediction models and the scheduling model building blocks

The three command buttons "Update trends," "Schedule!" and "Generate plan" were briefly mentioned in §8.2.1. These command buttons contain most of the functionality of the decision support system, since the first button functions as the initiator of computer procedures representing the prediction model building block, the second button functions as the initiator of the computer procedures representing the scheduling algorithm and the third button initiates the computer procedures that turns the solution from the scheduling algorithm into a legible schedule. The following subsections contain comprehensive explanations of the procedures associated with the command buttons.

The command button "Update Trends"

The command button "Update Trends" is used to separate the computationally demanding process of computing the parameter values (the P_{ij}, M_{ij}, R_{ij}, C_{ij} and $\bar{w}_j$ of Chapter 7) from the process of executing the scheduling algorithm, since the user may wish to rerun the scheduling algorithm without having to wait for the parameter values to be updated unnecessarily.

When the "Update trends" command button is left-mouse-clicked, the VBA code in Appendix A.1 is executed, here called the *UDT procedure*. An overview of the events that occur in this execution is shown in Figure 8.7(a), the components of which are described as follows:

Wait for user to click. The implementation waits for the user to click the command button before initiating the UDT procedure.

Initialise variables. The implementation initialises variables for the *number of varieties available* (NOV) by searching the top 998 rows of worksheet "V" and counting the number of entries in column "B", the *number of possible events* (NOE) by searching worksheet "EV" and counting the number of entries in column "B", the *number of toposequences* (NOS) by searching worksheet "S" and counting the number of entries in column "B," and, finally, the *number of field aspects* (NOFD) by searching worksheet "FD" and counting the number of entries in column "B." The name of the harvesting front (HF) that should be considered during the execution of a particular scheduling instance is captured from the user interface (the user must enter the name of the harvesting front for which scheduling is required), the *total number of fields* (TNOF) in the database variable is found by counting the entries from row 1 to row 999 in the worksheet "Start" and an initial estimate of the *number of fields* (NOF) to include in the schedule is found by evaluating a function that counts only those entries that have a harvesting front value that matches the name of the HF. The variable NOF is then adjusted for fields that have been harvested *completely.* The *first day of the year* (FDY) variable is captured from the user interface and the *number of periods* (NOP) variable is set to 52. Of the variables mentioned in this paragraph, only the variable NOF has previously been mentioned in this dissertation. The variable NOF corresponds to the size of the set I introduced in §6.1.

RV matrix. The RV matrix RVij is defined as a two-dimensional matrix with TNOF rows and NOP columns and is built up by starting with the topmost field in the database (the worksheet "FDB") and computing an RV %-value for the first period of the season. Subsequently, the RV %-values for the remaining periods are computed and these values become the first row of the matrix. This is repeated for all TNOF fields in the database. To compute each RV % value according to (7.10) of §7.4.3 and §7.4.4, the *field toposequence* (Sit), *field aspect* (FD), *cane variety* (V), regression model intercept (B0), regression model first-order coefficient (B1), regression model second-order coefficient (B11), event names for events 1–3 (EV1, EV2 and EV3), event dates for events 1–3 (EV1D, EV2D and EV3D), the percentages of the fields affected by events 1–3 (EV1P, EV2P and EV3P) are first captured for the relevant field from the worksheet "FDB". Secondly, the *Julian date* (JD) of the harvest of the field is determined and an RV % (RVBase) is calculated by applying the regression model to JD following the outline in §7.4.3. Thirdly, the effects of each event having taken place on each field is computed by means of (7.11) and (7.12) of §7.4.4. Here, $d^M_{0,\ell_i(k)}$ in (7.11) and $d^M_{1,\ell_i(k)}$ in (7.12) of §7.4.4 are called BEC0 and BEC1, respectively, and are located in worksheet "EV," in which they are listed by event name. The implementation then takes the relevant percentage of the field affected by each event H_{ik} of (7.11) and (7.12) of §7.4.4 (here denoted by EV1P, EV2P and EV3P), takes the event's interaction with cane variety $V^E_{i\ell_i(k)}$ (here denoted by VCoeff) from worksheet "V," takes the event's interaction with field toposequence $L^E_{i\ell_i(k)}$ (here denoted by SitCoeff) from worksheet "S", takes the event's interaction with field aspect $F^E_{i\ell_i(k)}$ (here denoted by FDCoeff) from worksheet "FD" and finally uses JD and the event dates (EV1D, EV2D and EV3D) to compute $q_{\ell_i(k)}$ of (7.9). The implementation may now by means of (7.9) compute the effects of the up to three events (here denoted by , RVEV1, RVEV2 and RVEV3, respectively) taken place on each field. RVBase is subsequently added to RVEV1, RVEV2 and RVEV3, to complete the computation of a single RV matrix entry, following (7.10). RVij is pasted into the worksheet "FDB" in the proper place.

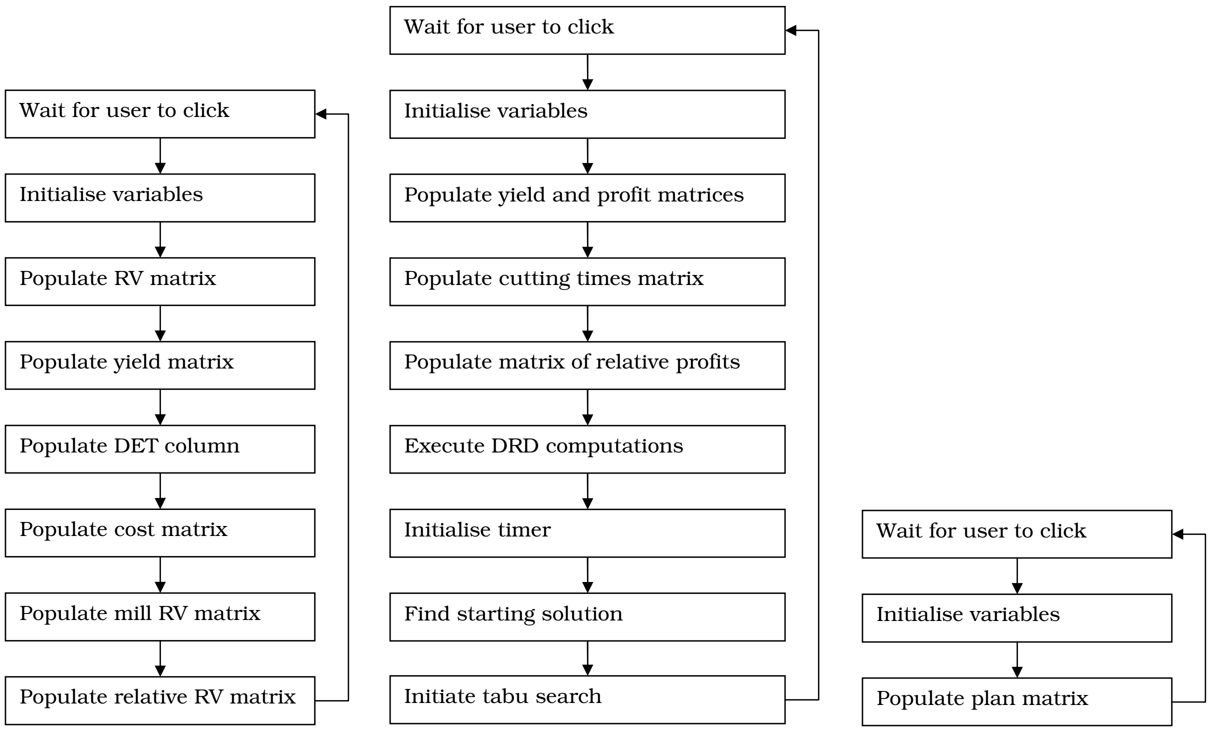
Yield matrix. The yield matrix Yij comprises TNOF rows and NOP columns, and is constructed in a similar fashion as RVij. Here (7.7), (7.8), (7.5), (7.6) of §7.4.2 and the outline in §7.4.1 are used in succession to compute each entry of this yield matrix. The main difference is that MAV is applied by the calculated yield value for each field and period, but since the regressor unit is measured in *effective growth time* (EGT)—defined as the time in months outside the period of June, July and August allowed for the field to grow— the calculation involves this rather complicated step for every field and period. As for the RV matrix, Yij is constructed by finding the regression model value for each field and adding to it the each field's events' yield effects (here denoted by YEV1, YEV2 and YEV3) and iterating this computation over all fields and periods. The resulting matrix is then pasted into the worksheet "FDB".

DET. The implementation computes a column of differences between the user's own estimate of the yield at ESTD and the Yij-element corresponding to each field (row) and the period in which ESTD falls (column). The DET column vector is pasted into the worksheet "Start" so that it is visible within the user interface.

Cost matrix. The cost matrix Cij also consists of TNOF rows and NOP columns, and is constructed by computing one element at a time. Each element is computed by adding the cut, load and transport costs from the user interface to the sum of the following three terms as outlined in §7.5 according to (7.15): (1) the wet probability of the field and period adjusted for toposequence and aspect multiplied by the zone loading cost from the user interface, (2) the difference between the time elapsed since the event and the allowable time window for the event (HTW) for each event (found in the worksheet "EV") multiplied by the event penalty from the user interface, and finally (3) the difference in time between the period and the harvest time date (HTD) (found in the worksheet "EV") multiplied by the event penalty from the user interface. Cij is subsequently pasted into the worksheet "FDB."

Mill RV matrix. The mill RV matrix (MRVij) is constructed by means of the relevant regression coefficients from worksheet "V" based on the choice of RV-Season from the user interface. Each period's value is calculated separately, but all fields in the same harvesting front are assumed to deliver to the same mill—therefore the matrix may be constructed by copying one row TNOF times. The matrix MRVij is subsequently pasted into the worksheet "FDB."

Relative RV matrix. The relative RV matrix (RRVij) is constructed element by element by subtracting the MRVij value from the RVij value and then adding the average mill RV value (the row average). RRVij is subsequently pasted into the worksheet "FDB."



(a) "Update trends." (b) "Schedule!" (c) "Generate plan."

Figure 8.7: *Flowcharts of the actions executed when the command buttons "Update trends," "Schedule!" and "Generate plan" are left-mouse-clicked.*

The command button "Schedule!"

The command button "Schedule!" provides the user with control over the scheduling algorithm. The command button is connected to the VBA procedure in Appendix A.2. The algorithm is described in Chapter 6 and is a tabu search employing a shift neighbourhood and an ejection chain. In the following description of the flow of the activities within the scheduling algorithm, the term *problem* should be interpreted as an instance of the scheduling problem described in §6.2 comprising a set of sugarcane fields and harvesting periods. The working procedure is illustrated in Figure 8.7(b).

Wait for user to click. When the user left-clicks the "Schedule!" button, the implementation initiates the code in Appendix A.2.

Initialise variables. Variables used in this procedure are initialised in such a way as to minimise memory usage and computation speed, without encroaching on the possibility to extend the procedure to larger problems (up to 999 fields).

Yield and profit matrices. The problem-specific yield and profit matrices (the Mij and Pij matrices, respectively) comprise NOF rows by NOP columns (note that NOP does not necessarily have the value 52 anymore) and are constructed by selecting the rows which correspond to the fields identified by the current HF and which have not been harvested completely. The selection occurs from the yield part of the worksheet "FDB" that covers the columns (periods) implied by the user input start and end of schedule. To compute a yield value for each field in tonnes (to populate the Mij matrix), the selected part of the worksheet "FDB" (which is given in tonnes per hectare) is multiplied row-wise by the field's area. To populate the Pij matrix, a profit value for each problem field is computed by multiplying the corresponding Mij matrix element by the difference between the RV price multiplied by the relative RV % (from the corresponding field and period on the relative RV % part of the worksheet "FDB") and the cost (from the corresponding field and period of the cost part of the worksheet "FDB"). The last computation follows that of (7.1) in §7.1.

Cutting times matrix. The cutting time matrix CTij requires the computation of the time that it takes to harvest a field during a certain time period and depends on the choice of method specified by the user via the user interface. The method may either be based on the area of the field or on the yield value of the field. In the former case, the computation is independent of the time period, and is thus straightforward. The cutting time for each field is computed by dividing the field area by the total problem area, subsequently multiplying that value by the schedule duration expressed in days (populating the CTAij matrix). However, if the latter method is chosen, the total problem yield is first computed by summing the average yield for each field. The cutting time for each field and each period is computed by dividing the regression model yield for that field and period by the total problem yield, and subsequently multiplying that value by the schedule duration expressed in days (populating the CTYij matrix). Both methods are approximations, since the time required to cut a field may depend on *both* area and yield. Despite the approximations, the objective function is non-linear, which becomes clear when considering that the yield value of a field depends on the harvesting time period, which in turn depends on the harvesting sequence of fields preceding the field in question.

Matrix of relative profits. The matrix of relative profits (Nuij) is used in the ejection chain part of the tabu search algorithm. The relative profit value is computed for each field and period by dividing the corresponding profit matrix value by the largest profit matrix value for the field. Thus each value in the matrix Nuij expresses the relative performance of a field during a particular period, compared to what the field achieves in the period of best performance.

DRD computations. The user interface shows two values of DSS estimated DRD, namely a DRD based on the maximum yield of each field, and a DRD based on the average yield of each field. The DRD based on maximum yield is found by summing the maximum yield of each field found throughout the Mij matrix and dividing that number by the duration of the season expressed in days (LOS). The DRD based on average yield is found by summing each field's average yield across the Mij matrix, and subsequently dividing that number by the LOS.

Initialise timer. The timer used to control the run-time of the scheduling algorithm is set equal to system time plus the user-specified allotted time from the user interface.

Starting solution. At this stage a pseudo-random starting solution is generated by generating a random permutation of a range of integers between 1 and NOF.

Tabu search. The starting solution is used as input to the tabu search algorithm. The best possible solution found during the tabu search is returned from this step and is stored in the worksheet "Sequence."

Field	Area	Accs	Vrty	Month	AgeBrn	TnHa	Tons	RV	RRV	MRV	Cost	Ev1	Ed1	Ep1	Ev2	Ed2	Ep2	Ev3	Ed3	Ep3	CutJD
F4	5.63	1	N40	Mar	13.2	97	548	10.5	11.7	10.9	R 66	Fire warm	03/17	100	Normal	01/01	0	Normal	01/01	0	76
F9	9.18	2	N16	Apr	21.4	100	917	11.8	11.9	12.0	R 76	Plough out early	03/17	100	Normal	01/01	0	Normal	01/01	0	99
F1	6.76	2	N16	May	23.2	100	677	12.2	11.2	13.1	R 76	Normal	01/01	0	Normal	01/01	0	Normal	01/01	0	136
F7	5.52	1	N16	Jun	23.2	101	556	12.4	10.9	13.7	R 61	Normal	01/01	0	Normal	01/01	0	Normal	01/01	0	163
F3	4.49	1	N35	Jul	16.1	104	467	13.9	12.1	13.9	R 60	Normal	01/01	0	Normal	01/01	0	Normal	01/01	0	185
F6	8.95	1	N40	Jul	22.4	102	914	14.0	12.1	14.0	R 60	Normal	01/01	0	Normal	01/01	0	Normal	01/01	0	203
F5	9.32	1	N12	Aug	25.5	101	939	14.0	12.2	14.0	R 67	Normal	01/01	0	Normal	01/01	0	Normal	01/01	0	239
F8	4.61	2	N12	Oct	25.1	103	474	13.8	12.4	13.5	R 76	Normal	01/01	0	Normal	01/01	0	Normal	01/01	0	277
F2	3.35	2	N16	Oct	29.1	97	325	12.5	11.6	13.0	R 76	Normal	01/01	0	Normal	01/01	0	Normal	01/01	0	295
F10	6.09	1	N12	Nov	20.8	114	696	13.2	12.8	12.6	R 66	Normal	01/01	0	Normal	01/01	0	Normal	01/01	0	309

Table 8.1: *An example of a ten-field harvesting plan constructed by the computer implementation of the DSS. Here the column* Field *contains the names of the fields in the schedule, the column* Area *contains the areas of the fields, the column* Month *contains the months of the scheduled fields' harvesting dates, the column* Accs *contains the access categories of the fields, the column* Vrty *contains the cane varieties of the fields, the column* AgeBrn *contains the ages of the fields at the time of the fields' scheduled burning, the column* TnHa *contains the cane yields per hectare for the fields at their times of scheduled harvest, the column* Tons *contains the cane yields of the fields at their times of scheduled harvest, the columns* RV, RRV *and* MRV *contain the RV percentages, the relative RV percentages and mill area RV percentages for the fields at their scheduled times of harvest, respectively, the column* Cost *contains the harvesting cost in R.ha*$^{-1}$ *at the fields' times of scheduled harvest, the column* Ev1 *contains the names of any first events having struck the fields, the column* Ed1 *contains the dates of any such events, the column* Ep1 *contains the percentages of the fields affected by these events, the column* Ev2 *contains the names of any second events having struck the fields, the column* Ed2 *contains the dates of the second events, the column* Ep2 *contains the percentages of the fields having been affected by such an event, the column* Ev3 *contains any third events having struck the fields, the column* Ed3 *contains the dates of any third events, the column* Ep3 *contains the percentages of the fields having been affected by the third events and the column* CutJD *contains the days of the year on which the fields are scheduled to be harvested.*

Factors	Low level	Centre point	High level
Tabu tenures [A edges]	1	5	9
MaxNoBigImpr [B iterations]	1	3	5
MaxNoImpr [C iterations]	1	10	19
Timelimit [D seconds]	10	40	70

Table 8.2: *The first verification experiment's factors and levels.*

The command button "Generate plan"

The command button "Generate plan" should be clicked upon completion of the procedures initiated after clicking on the "Schedule!" command button in order to produce a printable preview of the solution output, which is also stored in the worksheet "HarvestPlan." The working of the procedure is illustrated in Figure 8.7(c).

Wait for user to click. When the user clicks, VBA executes the procedure in Appendix A.3.

Initialise variables. Variables are first initialised.

Plan matrix. The report is generated by means of populating a matrix that is called "Plan." The matrix has NOF+1 rows (one additional row for the column headings) and 22 columns (one for each report option). The matrix is populated field by field, one column at a time, by extracting data from the worksheets "FDB" and "Start," using appropriate functions to manipulate the relevant data into proper format. Upon completion of the last field in the harvest sequence, the matrix is pasted into the worksheet "HarvestPlan". An example of a typical harvest plan is shown in Table 8.1.

8.3 Verification of the DSS implementation

The decision support system functionality was verified by means of a number of design experiments described in this section. However, before any experiment was performed, the DSS was debugged and equipped with a random farm generator as well as a routine used to record relevant information about the progress of the scheduling algorithm itself.

8.3.1 Verification experiment 1: 20 fields

The first verification experiment was a 2^4 factorial design experiment in two replicates with six centre points (see, for example, [156]). The factors and levels used are shown in Table 8.2, where "Tabu tenures" denotes the lengths of any of the six tabu lists, "MaxNoBigImpr" denotes the number of ejection chains allowed in a row between improvements to the overall best objective function value before a new random starting solution is generated, "MaxNoImpr" denotes the number of iterations in a row between improvements to the best objective since the last randomly generated solution allowed before an ejection chain is generated and "Timelimit" denotes the time duration after which the algorithm terminates and returns the best solution found so far. Each of the 38 runs was a 20-field randomly generated farm with up to one event per field. The likelihood of an event occurring was set to 20 percent.

Treatment	A	B	C	D	R1 *% OPT*	R2 *% OPT*	R1 *ITR*	R2 *ITR*	R1 *RND*	R2 *RND*	R1 *EJC*	R2 *EJC*	R1 *BRK*	R2 *BRK*
1	1	1	1	10	91.79	85.07	482	475	14	14	17	17	4	10
2	9	1	1	10	92.53	92.28	385	398	18	19	20	21	2	3
3	1	5	1	10	91.78	92.1	479	489	7	5	39	30	3	4
4	9	5	1	10	89.78	91.12	380	392	7	6	41	36	5	8
5	1	1	19	10	90.55	89.69	500	470	6	6	8	7	3	6
6	9	1	19	10	92.01	90.73	251	266	3	4	4	4	4	2
7	1	5	19	10	91.72	90.3	480	497	1	1	15	14	3	10
8	9	5	19	10	77.04	92.16	251	262	0	0	7	8	4	2
9	1	1	1	70	92.32	92.21	3538	3491	108	105	110	107	70	84
10	9	1	1	70	90.97	92.64	2718	2673	169	164	171	165	6	11
11	1	5	1	70	91.68	92.84	3414	3486	40	44	214	227	78	71
12	9	5	1	70	91.73	91.59	2600	2661	59	65	302	330	13	18
13	1	1	19	70	91.72	89.39	3470	3475	49	47	50	51	45	63
14	9	1	19	70	52.89	91.4	1691	1738	30	31	30	33	7	6
15	1	5	19	70	93.02	90.04	3579	3541	18	17	92	87	11	6
16	9	5	19	70	90.34	92.91	1697	1687	10	10	55	55	6	6
17	5	3	10	40	86.25	78.11	1635	1615	13	12	39	41	119	108
18	5	3	10	40	90.6	91.96	1697	1696	13	12	41	41	124	125
19	5	3	10	40	91.14	92.73	1754	1681	11	13	37	44	132	93

Table 8.3: *The results of the various responses along with settings for each particular run for the first verification experiment. Here,* treatment *means a particular setting of the factor levels,* A, B, C *and* D. R1 *means the first replicate of a particular treatment and* R2 *means the second replicate of that treatment. Furthermore, % OPT denotes the percentage of the upper bound achieved by the objective function value of each replicate and treatment, ITR denotes the number of iterations completed by the algorithm, RND denotes the number of randomly generated starting solutions generated throughout the run, EJC denotes the number of ejection chains applied and BRK denotes the number of times the search escaped from a local optimum without using a random restart or ejection chain.*

The results of Verification Experiment 1 may be seen in Table 8.3, where "Run" denotes the number of the experimental row, which includes two separate experimental runs, "A" denotes the value or level of factor A for each experimental row, "B", "C" and "D" are the respective levels taken by factors B, C and D for each experimental row. In the table, "*% OPT*" (one column for each replicate) denotes the solution objective function values expressed as percentages of the upper bound, "*ITR*" denotes the total number of iterations completed during the search, "*RND*" denotes the number of randomly generated solutions used throughout the search, "*EJC*" denotes the number of ejection chains used throughout the search and "*BRK*" denotes the number of times that the tabu search found a locally better solution (between two execution-wise adjacent ejection chains) while being kept away by the tabu lists from the current local optimum. A *BRK* value of 10 means, for example, that ten times during the search, the algorithm first found a local optimum and was disallowed from revisiting it by the tabu lists, but while being disallowed to revisit the local optimum, found a solution that was better than the local optimum.

The results of Table 8.3 were analysed by means of an ANOVA using backward elimination regression for factor selection. For each response, the starting model was a full factorial design with interaction effects[3], and the results for the backward elimination regression with an alpha-out at 0.1 of the five responses are shown in Table 8.4.

The quality of solutions, as measured by the optimality gap, clearly decreases as the tabu tenures

[3]Interaction effects between factors, for example, the AC interaction effect denoting the interaction effect between factor A and factor C, represent the occurrence of a dependency in that the effect of either one of the factors on the measured response depends on the concurrent value of the other variable.

Res		A	B	C	D	AB	AC	AD	BC	BD	CD	ABC	ABD	ACD	BCD	ABCD	Cur
	Mod	Yes	No	Yes	No	No	Yes	No	No	Yes	No	No	Yes	No	Yes	Yes	
% OPT	Effect	−7.2	2.8	−6.7	−2.8	2.3	−6.5	−3.5	3.5	6.9	−3.0	2.7	7.1	−3.5	6.2	6.0	
	p-value	0.04	0.4	0.06	0.4	0.5	0.07	0.3	0.3	0.05	0.4	0.4	0.05	0.3	0.08	0.09	0.7
	Mod	Yes	No	No	Yes	No	No	No	No	No	No	No	No	No	No	No	
ITR	Effect	−746	−19	−260	2437	−10	−289	−578	43	−12	−199	−11	−14	−218	46	−16	
	p-value	0.05	1.0	0.5	0.00	1.0	0.4	0.1	0.9	1.0	0.6	1.0	1.0	0.5	0.9	1.0	0.9
	Mod	No	Yes	Yes	Yes	No	No	No	No	No	No	No	No	No	No	No	
RND	Effect	6.6	−32	−38	53	−4.1	−14	6.6	17	−25	−29	7.4	−3.6	−12	14.6	5.9	
	p-value	0.7	0.1	0.05	0.007	0.8	0.4	0.7	0.4	0.2	0.1	0.7	0.8	0.5	0.4	0.8	0.3
	Mod	No	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes	No	No	Yes	No	No	
EJC	Effect	10.6	44.4	−82	109	0.6	−28	12	−25	31	−61	−5.9	1.9	−24	−17	−5.1	
	p-value	0.4	0.001	0.00	0.00	1.0	0.03	0.3	0.05	0.02	0.00	0.6	0.9	0.06	0.2	0.7	0.01
	Mod	Yes	No	No	Yes	No	No	Yes	No	No	No	No	No	No	No	No	
BRK	Effect	−21	−2.3	−12	26	4.5	11	−22	−6.5	−2.8	−12	3.8	3.5	11	−6	4.8	
	p-value	0.02	0.8	0.1	0.004	0.6	0.2	0.01	0.4	0.7	0.1	0.7	0.7	0.2	0.5	0.6	0.00

Table 8.4: *Abbreviated ANOVAs of the five responses subject to backward elimination regression.* Res *denotes response,* Cur *denotes the quadratic curvature of the model and* Mod *represents the question of whether the coefficient should be included in the model or not.*

(factor A) increase. The between-ejection chain no-improvement iterations—local iterations—(factor C) may also have a negative effect on solution quality. The interaction effects that are significant are hard to explain due to lack of hierarchy; however, the AC-interaction effect is statistically significant, which would imply that the negative effect of tabu tenure on solution quality worsens at high levels of local iterations allowed (factor C). The BD-interaction effect is also statistically significant which implies that the time limit (factor D) plays a more important role if the number of ejection chains between random restarts (factor B) is larger, which makes sense since in that case the algorithm would spend more time semi-locally (within the same random starting point, but employing ejection chains for diversification) and would not diversify enough. The fact that the ABD-interaction effect is statistically significant implies that the BD-interaction effect may be larger at high levels of factor A, which could be interpreted as the inhibiting effect (on diversification) of increasing the tabu tenure may increase the reliance on time for diversification.

The number of iterations, *ITR*, decreases as the tabu tenures increase and, of course, increases as the time limit increases.

The number of random restarts, *RND*, decreases as the number of local iterations allowed is increased and decreases as ejection chains allowed increases, while it increases as the time limit is increased.

The number of ejection chains performed by the tabu search increases when the number of ejection chains allowed is increased (as expected) and decreases as the number of local iterations is increased (probably due to the additional search time spent between ejection chains). Increasing the time limit increases the number of ejection chains and the positive effect of increasing the number of ejection chains allowed is smaller at high levels of allowed local iterations (BC-interaction effect). Increasing the time limit also increases the positive effect of the number of allowed ejection chains (BD-interaction effect).

The number of "breakaways", *BRK*, seem to decrease with the tabu tenures, but increase with an increased time limit. The AD-interaction effect for this response implies that if one is interested in increasing *BRK*, one should use short tabu lists and a large time limit.

In order to further verify that the DSS functioned as intended, the best harvest schedule produced by each experimental run was examined. The main test performed was to ensure that no obvious, trivial improvements could be made by inspection, which would be the case if a field that was affected by a serious event were to be harvested too late. In addition to the harvest schedule, the DSS was made to plot a graph of information collected during each run. The

Treatment	Tabu tenure	Repeated measurements 1	2	3	4	Average
1	1	0.9449	0.9445	0.9446	0.9446	0.9446
2	2	0.9443	0.9444	0.9443	0.9445	0.9444
3	4	0.9446	0.9448	0.9446	0.9440	0.9445
4	12	0.9445	0.9446	0.9451	0.9440	0.9445
5	20	0.9444	0.9448	0.9440	0.9443	0.9444

Table 8.5: *Optimality gaps encountered during the second verification experiment.*

Source of variation	SS	df	MS	p-value
Between treatments	1.94×10^{-7}	4	4.85×10^{-8}	0.731
Within treatments	1.43×10^{-6}	15	9.54×10^{-8}	

Table 8.6: *Single-factor ANOVA of the optimality gaps in Table 8.5 encountered during the second verification experiment. Here* SS *denotes sum of squares,* df *denotes degrees of freedom and* MS *denotes mean square.*

graph plots the current objective function value, the best local objective function value (since the last random restart) and the best overall objective function value along with the number of iterations since the last ejection chain and the number of iterations since the last random restart. Both the harvest schedule and this graph are shown in Figures 8.8 and 8.9 for the second replicate of the first row of the experiment in Table 8.3.

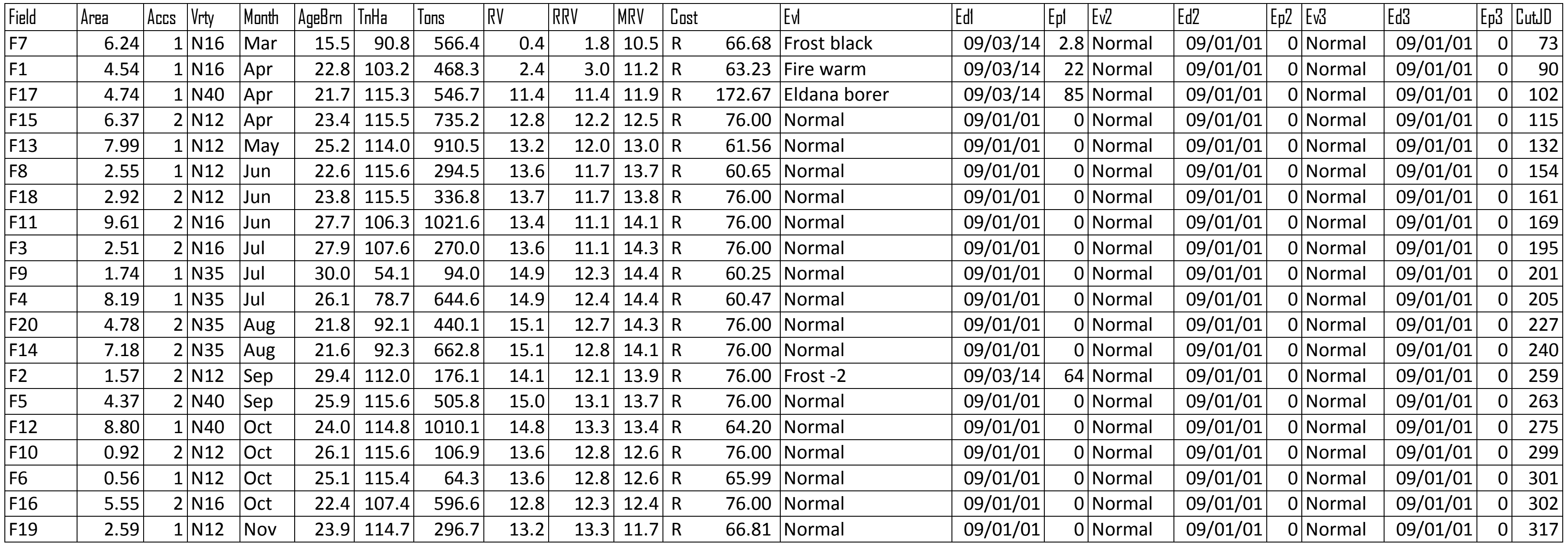
8.3.2 Verification experiment 2: 60 fields

Verification Experiment 2 was conducted as a single factor experiment where the only factor varied was the tabu tenures, which were set equal for all six tabu lists. Each run consisted of solving an instance four times (repeated measurements), and recording the average $\%\ OPT$ value and standard deviation over the four repeated measurements. The maximum number of iterations without improvement between ejection chains was set to ten since at this stage it had not been realised that the length of short-cyclings for this problem are longer than ten with the tenures employed in this experiment. The maximum number of ejection chains without any improvement to the best objective function value found so far was set to three so as to only allow a presumably low number of random restarts, while the time limit was set to sixty seconds. The results of this experiment are shown in Tables 8.5 and 8.6.

The results show that for this instance, the tenure does not influence solution quality (the p-value for the between-treatment effects is 0.731, which is not less than 0.05, a common significance level threshold). An ANOVA was also conducted on the number of iterations completed when attempting to solve each problem instance, the results of which are shown in Table 8.7. The results clearly indicate that a large tabu tenure significantly slows down the algorithm, as expected.

Source of variation	SS	df	MS	p-value
Between treatments	813.7	4	203.425	4.55×10^{-7}
Within treatments	99.25	15	6.61667	

Table 8.7: *Single-factor ANOVA of ITR-values for the second verification experiment. Here* SS *denotes sum of squares,* df *denotes degrees of freedom and* MS *denotes mean square.*



Field	Area	Accs	Vrty	Month	AgeBrn	TnHa	Tons	RV	RRV	MRV	Cost		Ev1	Ed1	Ep1	Ev2	Ed2	Ep2	Ev3	Ed3	Ep3	CutJD
F7	6.24	1	N16	Mar	15.5	90.8	566.4	0.4	1.8	10.5	R	66.68	Frost black	09/03/14	2.8	Normal	09/01/01	0	Normal	09/01/01	0	73
F1	4.54	1	N16	Apr	22.8	103.2	468.3	2.4	3.0	11.2	R	63.23	Fire warm	09/03/14	22	Normal	09/01/01	0	Normal	09/01/01	0	90
F17	4.74	1	N40	Apr	21.7	115.3	546.7	11.4	11.4	11.9	R	172.67	Eldana borer	09/03/14	85	Normal	09/01/01	0	Normal	09/01/01	0	102
F15	6.37	2	N12	Apr	23.4	115.5	735.2	12.8	12.2	12.5	R	76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	115
F13	7.99	1	N12	May	25.2	114.0	910.5	13.2	12.0	13.0	R	61.56	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	132
F8	2.55	1	N12	Jun	22.6	115.6	294.5	13.6	11.7	13.7	R	60.65	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	154
F18	2.92	2	N12	Jun	23.8	115.5	336.8	13.7	11.7	13.8	R	76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	161
F11	9.61	2	N16	Jun	27.7	106.3	1021.6	13.4	11.1	14.1	R	76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	169
F3	2.51	2	N16	Jul	27.9	107.6	270.0	13.6	11.1	14.3	R	76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	195
F9	1.74	1	N35	Jul	30.0	54.1	94.0	14.9	12.3	14.4	R	60.25	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	201
F4	8.19	1	N35	Jul	26.1	78.7	644.6	14.9	12.4	14.4	R	60.47	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	205
F20	4.78	2	N35	Aug	21.8	92.1	440.1	15.1	12.7	14.3	R	76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	227
F14	7.18	2	N35	Aug	21.6	92.3	662.8	15.1	12.8	14.1	R	76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	240
F2	1.57	2	N12	Sep	29.4	112.0	176.1	14.1	12.1	13.9	R	76.00	Frost -2	09/03/14	64	Normal	09/01/01	0	Normal	09/01/01	0	259
F5	4.37	2	N40	Sep	25.9	115.6	505.8	15.0	13.1	13.7	R	76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	263
F12	8.80	1	N40	Oct	24.0	114.8	1010.1	14.8	13.3	13.4	R	64.20	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	275
F10	0.92	2	N12	Oct	26.1	115.6	106.9	13.6	12.8	12.6	R	76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	299
F6	0.56	1	N12	Oct	25.1	115.4	64.3	13.6	12.8	12.6	R	65.99	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	301
F16	5.55	2	N16	Oct	22.4	107.4	596.6	12.8	12.3	12.4	R	76.00	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	302
F19	2.59	1	N12	Nov	23.9	114.7	296.7	13.2	13.3	11.7	R	66.81	Normal	09/01/01	0	Normal	09/01/01	0	Normal	09/01/01	0	317

Figure 8.8: *Row 1, second replicate best harvest plan (screen-shot).*

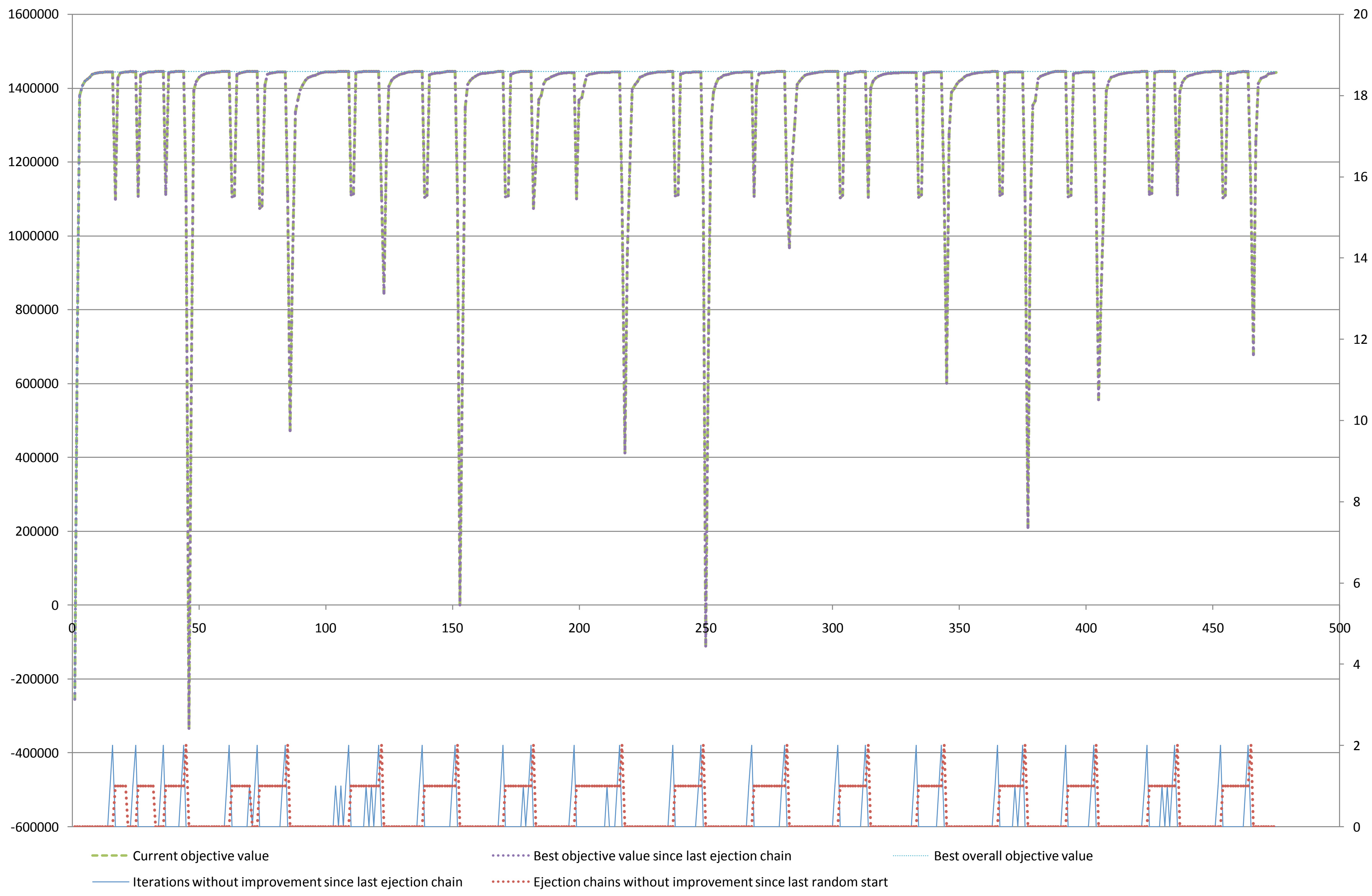


Figure 8.9: *Row 1, second replicate solution history. The dashed lines at the bottom of the graph are plots of the number of ejection chains (values read off the right-hand-side vertical axis) since the last local improvement, the lines at the top of the graph are plots of the objective function value (values read off the left-hand-side vertical axis).*

	Repeated measurements								
15 minutes	0.8856	0.8828	0.8881	0.8882	0.8872	0.8854	0.8885	0.8863	0.8867
60 minutes	0.8894	0.8885	0.8888						
120 minutes	0.8899								

Table 8.8: *Values of % OPT corresponding to the third verification experiment. The 120 minute run is not part of the ANOVA; it was only run for comparison.*

Source of variation	SS	df	MS	p-value
Between treatments	1.2856×10^{-5}	1	1.2856×10^{-5}	0.05
Within treatments	2.5982×10^{-5}	10	2.59817×10^{-6}	

Table 8.9: *Single-factor ANOVA of % OPT values for the third verification experiment. Here* SS *denotes sum of squares,* df *denotes degrees of freedom and* MS *denotes mean square.*

8.3.3 Verification experiment 3: 200 fields

A few tabu search runs were finally performed on a single problem instance involving two-hundred fields with an allotted solution time of fifteen minutes, while a few runs were conducted with a solution time of sixty minutes, and there is indication that the solutions obtained after fifteen minutes are significantly worse than those obtained after sixty minutes. The *% OPT* values obtained are shown in Table 8.8.

The results of the corresponding ANOVA are shown in Table 8.9 and indicate with a p-value of 0.05 that there is a significant difference in solution quality between the 15 minute and 60 minute time limit solutions uncovered for the 200 field instance at hand, while the 120 minute run indicates that further improvement in objective function value is possible.

8.4 Determining the tabu list tenures

The problem under study displays an interesting structure. If, for example, there are two hundred fields to be harvested during a period of forty weeks, some neighbouring pairs of fields are certainly harvested during the same week. If, in this example, all fields are equally large, it is easy to see that five fields are harvested each week. There are several shift moves in this example that will not produce a change in objective function value, since each week allows for one-hundred-and-twenty possible different arrangements of the five fields. Even in a situation where fields are not equally large, there may be many possible shift moves that do not cause a change in objective function value. A solution that has neighbouring solutions—direct neighbours or neighbours' neighbours and so on—with the same objective function value is said to lie on a *plateau* in the search space. Plateaux may become large if there are many fields in comparison to the length of the harvesting periods, and in the above example, the number of fields on each plateau is at least[4] 120^{40}. Discovery of plateaux in the search space is certainly not new; in early papers on tabu search methods non-improving moves were allowed to occur if there was no improving neighbour of the current solution available [73]. However, if there is a large plateau at a local optimum, one might argue that it is highly unlikely that the search would return to a previously visited solution, even if the tabu tenure is zero or one. In the above example, each solution has 800 direct neighbours on the same plateau, so moving from

[4]There may be other shift moves than the ones "within" weeks that do not cause a change in objective function value.

the current solution to another solution when at a local optimum implies a 1/800 probability of returning to the current solution at the next iteration if the tabu tenure is zero. If the tabu tenure is 1, the probability is $1/(6.4 \times 10^5)$. This suggests that tabu tenures need not be large for this scheduling problem.

However, the situation is different for instances with a relatively small number of fields compared to the number of harvesting periods. Assuming that there are problem instances for which there are no plateaux, consider a situation in which the tabu search has reached a local optimum. The first shift move taken will be a move that causes the least possible decrease in objective function value. The new solution is worse than the previous solution. If the new solution has a neighbour which is better than the previous solution, the tabu search will move to the neighbour of the new solution. If there is no such neighbour, the tabu search will attempt to move back to the previous solution, and the only manner by which to prevent this type of cycling is by classifying the move back to the previous solution as tabu. Other than running tests, there is no way of computing how long a tabu list should be in order to prevent this kind of “short-cycling” behaviour.

An experiment designed to determine the necessary tabu tenures consisted of running several instances of a problem with forty harvesting periods and twenty fields, all ranging between three and nine hectares in size, which makes it unlikely that there are plateaux present in the solution space. Each instance was repeatedly solved, using different tenures. The search time limit was set to thirty seconds, the maximum number of non-improving iterations between ejection chains was set to 100 and the number of ejection chains allowed without improvement was set to 10 so that there would not be any random restarts. The solution runs were examined for instances of short-cycling, and the length of the shortest cycle was recorded as well as the *% OPT* and *BRK* values. The results are shown in Table 8.10.

	20-Field instances					
	Instance 1			Instance 2		
Tenure	*% OPT*	Cycle length	*BRK*	*% OPT*	Cycle length	*BRK*
1	94.33272	4	4	92.71313	4	18
	94.33272	4	8	92.71313	4	12
2	94.33272	6	17	92.71313	6	25
	94.33272	6	22	92.71313	6	24
3	94.33272	8	24	92.71313	8	36
	94.33272	8	35	92.71313	8	35
4	94.33272	10	38	92.71313	10	54
	94.33272	10	36	92.71313	10	34
5	94.33272	12	34	92.71313	12	31
	94.33272	12	40	92.71313	12	22
6	94.31389	1	6	92.71313	1	1
	94.32819	1	5	92.71313	1	2
7	94.33272	1	7	92.71313	1	1
	94.33272	1	4	92.71313	1	2

Table 8.10: *Results of different tenures on the optimality gap, cycle length and BRK on two 20-field problem instances. Cycle length denotes the number of iterations completed by the tabu search (at least) before returning to a previously visited local optimum. There were 40 time periods and fields were between six and nine hectares in size; this size distribution coupled with the 20/40 ratio of the number of fields to the number of periods lead to a problem instance containing very few plateaux.*

The results of this experiment show that for the two instances the largest number of *BRK*s were obtained for tabu tenures of three to five. Tabu tenures of six or seven led to a complete exhaustion of the shift neighbourhood, which then returned the input solution as output. An exhaustion of the shift neighbourhood occurred once when the tenure was equal to five.

The next tabu tenure experiment was performed on two forty-field problem instances with tabu tenures between four and twelve. The results are shown in Table 8.11 and indicate that there is little difference in the *% OPT* value obtained using the different tabu tenures. At a tabu tenure of twelve, there were cases of exhaustion of the shift neighbourhood.

The results suggest that a cycle will be at least as long as two plus the tabu tenure multiplied by two. One may thus draw the conclusion that the number of iterations between ejection chains should be allowed to be at least the length of a cycle, but in the case of problems with small or no plateaux, not much more. In the case of problems with large plateaux, the tabu list need not be larger than 1 or 2, but it may be beneficial to allow the number of iterations between ejection chains to exceed these values substantially, in order to allow the search to explore the plateaux in search of possible escape routes towards better solutions.

8.5 Chapter summary

In this chapter, the decision support system as a concept was described in terms of an architectural framework into which building blocks may be incorporated to create a functioning whole. The DSS architectural framework was also implemented on a personal computer, using a programming language available to anyone with access to Microsoft Excel 2007. The computer implementation of the DSS was described by means of screen-shots and bulleted descriptions of the various parts of the user interface and the procedures embedded within. The DSS computer implementation was thoroughly verified by means of different problem instances and parameter settings were discussed at some length, concluding that the number of iterations between ejection chains allowed should be at least as many as twice the tabu tenure plus two. The tabu tenure should be approximately 3–4 for normal 20-field instances, 9–10 for normal 40-field

	40-Field instances					
	Instance 1			Instance 2		
Tenure	*% OPT*	Cycle length	*BRK*	*% OPT*	Cycle length	*BRK*
4	92.96646	10	13	92.71313	10	10
5	92.95090	12	8	92.71313	12	9
6	92.95046	14	19	91.72778	14	4
7	92.96168	16	25	91.75036	16	10
8	92.94984	18	7	91.75510	18	10
9	92.96172	20	17	91.76673	20	11
10	92.96517	22	7	91.77277	22	14
11	92.95375	24	12	91.75299	24	9
12	92.96168	1	6	91.75423	1	1

Table 8.11: *Results of different tenures on the optimality gap, cycle length and BRK on two 40-field problem instances. Cycle length denotes the number of iterations completed by the tabu search (at least) before returning to a previously visited local optimum. There were 40 time periods and fields were between five and seven hectares in size; this size distribution coupled with the 40/40 ratio of the number of fields to the number of periods lead to a problem instance containing a number of small plateaux.*

instances and may be set slightly larger for larger problem instances, but that the number of breakaways from local optima (the desired effect of having tabu lists) do not seem to increase beyond a certain size tenure, while computational time does increase.

This chapter, together with Chapters 6 and 7 complete the fulfilment of Dissertation Objectives VI and VII of §1.3.

CHAPTER 9

Validation of the DSS

Contents

This chapter contains a description of the results of a validation experiment conducted in order to test the desirability of the decision support output. The validation experiment was started in March 2009 using a development version of the decision support system, earlier referred to as the DSSDV, whose scheduling model building block was based on the BMF of §6.1 and an early implementation of the SMF of §6.2, which had not been thoroughly verified at that stage. The validation was performed in a *medium-scale commercial harvesting group* context, and is thus also valid for the *medium-scale solitary commercial grower* context. The final DSS architecture and implementation presented in §8.1 and §8.2 was based on the SMF of §6.2. The SMF introduced into the DSSDV in June 2009 started generating parallel schedules on June 9. The experiment continued throughout the 2009 harvesting season and was restarted at the onset of the 2010 harvesting season, this time using the full DSS described in Chapter 8. In this experiment the DSS was used to generate two schedules for each of two harvesting fronts involving a number of farms in the South African province of KwaZulu-Natal.

9.1 Validation of the DSSDV

The DSSDV was designed in Microsoft Excel, as was the DSS, but the algorithms used to solve the scheduling model in the DSSDV were programmed in Wolfram's Mathematica as opposed to VBA For Excel which was used in implementing the full DSS. The schedules of the DSSDV were generated using a Dell Dimension 1.7 GHz Intel Celeron with 1 GB RAM.

The DSSDV was evaluated during the 2009 sugarcane harvesting season at four farms in the Eston Mill area in KwaZulu-Natal. Data from the four farms were first used to populate the models necessary to predict the yield and RV % of the varieties present on the farms. The DSSDV was then populated with these models together with the field data from the farms, and was subsequently used to generate harvesting schedules throughout the season, approximately once every two weeks. The manager of the harvesting front was responsible for returning a simple form containing answers to the questions: (1) "*Which schedule did you prefer?*", (2) "*Which fields did you harvest this* [period]*?*", (3) "*Grade the schedule that you preferred from 1–5, where 1 is very poor, 2 is poor, 3 is* [acceptable]*, 4 is good and 5 is very good.*", (4) "*Write any comments you may have regarding the schedules or anything that might relate to this past* [period's] *harvest planning.*" The resulting form, called the "*weekly harvest schedule return form*", was returned in conjunction with receiving new schedules. Apart from the simple formality of the form, numerous phone-calls to the manager and the growers were made by the author towards gaining a fuller understanding of any issue or viewpoint arising during the validation and development work.

The four growers participating in the validation project had previously formed a syndicate, and had previously employed a single harvesting front approach towards managing their harvesting operation. However, the growers decided to split the harvesting operation into two harvesting fronts, called HF A and HF B respectively, consisting of the fields belonging to two growers each. This decision was taken by the growers in order to remedy the previously unfair usage *vs.* internal costing of mechanised equipment. The unfairness arose due to differences in usage of certain equipment, which, in turn, was due to differences in topography between the two harvesting fronts. In the following account of the 2009 harvesting season, each harvesting front is treated separately.

9.1.1 Harvesting front A

Geographically, HF A is situated a few kilometres north of Richmond, KwaZulu-Natal. The *area under cane*[1] for HF A is approximately one hundred-and-fifty hectares spread across thirty-seven fields of between 0.56 and 9.18 hectares each. The varieties present in this harvesting front at the beginning of the 2009 milling season—which commenced on 18 March 2009—were N12, N16, N35, N37 and N41. The two growers connected to HF A delivered all of their cane to Eston Mill, located approximately thirty-five kilometres from the centre of the harvesting front. In 2008, Eston Mill crushed 1.51 million tonnes of sugarcane—a slight increase from 2007, 2006 and 2005 when it crushed 1.46, 1.35 and 1.40 million tonnes, respectively. The mill was commissioned by Illovo in 1995, and by 1996 it was crushing 1.11 million tonnes of sugarcane annually. All of the cane from the area is delivered by road, as is the produced sugar to Durban.

Table 9.2 shows all the schedules that were sent to the manager of HF A during the season and Table 9.5 contains the growers' grades and comments and the lessons learnt from each schedule.

Two measures, called the *scheduling prediction desirability 1* (SPD-1) and *scheduling prediction desirability 2* (SPD-2), were formulated in order to provide a numerical means by which to judge the desirability of the various schedules. For a particular schedule, SPD-1 is defined as the percentage of the fields scheduled for period 1 and period 2 combined (the first month of the schedule), that were actually harvested during each field's respective period or during the two months following each field's respective period. As an example of computing an SPD-1 value, consider the first schedule of HF A, which is the schedule listed in the column labelled

[1]The area that is considered to be up for harvest during one particular season.

	Harvesting front			Harvesting front	
Schedule	HF A	HF B	Schedule	HF A	HF B
BMF1	88	96	—	—	—
BMF2	85	88	—	—	—
BMF3	90	104	—	—	—
BMF4	68	62	—	—	—
BMF5	67	103	—	—	—
BMF6	—	101	SMF6	42	69
BMF7	—	80	SMF7	51	77
BMF8	—	49	SMF8	43	59
BMF9	—	32	SMF9	29	29
BMF10	—	50	SMF10	39	65
BMF11	—	52	SMF11	41	52

Table 9.1: *SPD-2 values for all schedules generated throughout the validation of the DSSDV. A paired Student's t-test for comparison of means shows no significant difference in SPD-2 values between the schedule set BMF6–BMF11 and the schedule set SMF6–SMF11. Another paired t-test shows, however, that the SMF schedules have a lower SPD-2 value for HF A than for HF B. There was no significant such difference between the BMF schedules of HF A and HF B.*

"BMF1" in Table 9.2. The first period of this schedule is the period 18/3–30/3, which contains the fields O32A, O41 and T216. The two following months comprise the four periods 31/3–14/4, 15/4–26/4, 27/4–10/5 and 11/5–25/5. The column labelled "Actual" is now searched over the five periods 18/3–30/3 to 11/5–25/5 and the number of occurrences of the fields O32A, O41 and T216 is counted. If some field appears more than once, only one occurrence of that field is recorded. In this example, the only such occurrence is that of field T216 during the period 11/5–25/5, so the number of occurrences during this computation so far is one. The second period of the column "BMF1" contains field T225, and the field occurs four times during the "Actual" column periods 31/3–14/4, 15/4–26/4, 27/4–10/5, 11/5–25/5 and 26/5–8/6, which is only counted as one occurrence (the harvest of this field occurred in portions spread across a period of more than one month). In summary, there are two occurrences of fields from the first month of the schedule in question present in the following periods of the actual harvesting sequence, and there are four fields in total scheduled during the first two periods, so the SPD-1 value is $2/4 \times 100$ % = 50 %.

The second measure (SPD-2) is the average time difference in days between scheduled day and actual harvesting day for the ten first fields of a schedule. For example, the SPD-2 for the column "BMF1" in Table 9.2 is the average of the values 156, 69.5, 55, 0, 41, 83.5, 114.5, 152.5, 55 and 152.5, which yields an SPD-2 of 87.95. SPD-2 values for all schedules and both harvesting fronts are presented in Table 9.1.

The first schedule (HF A BMF1) of the season for HF A was generated on 18 March 2009 but was cancelled and regenerated on 20 March due to an error in computing the values of R_{ij}. The solution to the BMF is within 1.6 % of optimality and the corresponding local search time limit was 15 hours. The grower response to the first schedule is clear in that the schedule does not reflect the difference in maturity between varieties properly. The growers argued that N35 and N37 are relatively more mature early in seasons preceded by cold and wet conditions compared to N12 and N16, and should thus be given precedence for harvesting. By 31 March 2009, field O4AB had been split into two fields—O4A and O4B—and O4A had been harvested during the period 18–30 March 2009. The fields recommended for harvesting by HF A BMF1 for the first month are O32A, O41, T216 and T225. The harvesting of T225 was actually begun within two weeks, T216 was actually harvested within two months, O41 was actually harvested within three months and O32A was actually harvested within five months. The SPD-1 value for this

		HF A DSSDV Schedules										
P	Actual	BMF1	BMF2	BMF3	BMF4	BMF5	SMF6	SMF7	SMF8	SMF9	SMF10	SMF11
18/3–30/3	O23 O26 O4A O8	O32A O41 T216	—	—	—	—	—	—	—	—	—	—
31/3–14/4	T225	T225	T225	—	—	—	—	—	—	—	—	—
15/4–26/4	T225	T231 T235	T232	O31 T203 T235								
27/4–10/5	T225	O40	O34 T223	T225	T235 T225	—	—	—	—	—	—	—
11/5–25/5	T225 T216 O17	O15 O23 O46	O17 O46 T235	O41 T223	O32A T216 T225B	O36	—	—	—	—	—	—
26/5–8/6	O42 O41 T231	O19 O42	O3C T231	O2A O32A O32B O46 O47	O19 O29 O3C O46 O4B	T232	—	—	—	—	—	—
9/6–22/6	T216 T225B T231 O16 O29	O2A T216B	O2A T228	O17 O4B	O12 O31	O1A	O46 O29 O3C T235 O32A O19 O16	—	—	—	—	—
23/6–6/7	O19 O13	O36	O32B O47 T225B	O13 T228	O47 T231	T215	O13 T234	O3C O32B O46 O32A T203 T235	—	—	—	—
7/7–21/7	T216 T228 T235 T234 O13	O1A	O1A	O1A	T232	O1B O42	T231 T216 T203	O19 O13 T234	O46 T235 T203 O32B O32A O3C	—	—	—
22/7–4/8	T216B O1A O1B O2A	O13 O29 O3C	T215	O12 O1B O29	O1B O2A	O12 O41	O36 T216B	O36 O4B	O36 T228	O46 O36	—	—
5/8–19/8	O2A O36	O16 O17	O29 O4B T216B	O3C T231	O41 T223	O40	T225B T228 O2A	T216B T228	O2A T216B	O3C O32B T203 O1B	O46 O3C O36	—
20/8–3/9	T223 T215 O40 T203 O47 O32A O31	O1B O47 T203	O12 O1B	O34 T216B	O13 T228	O31 O3C	O1B	O1B O47 O12	O1A	T216B O32A O31	T203 O17	O3C O23 O46 O32A O34
4/9–30/11	O12 O15 O32B O34 O3B O3C O46 O4B T232	T232 O26 O32B T228 O12 T225B O31 O34 O8 O4AB T223 T215	O36 O16 O41 O19 O32A O13 O42 O15 T216 O40 O31 T203	O15 T216 T232 O42 T225B O40 T215 O36 O16 O19	O15 T216B O17 O42 O1A O16 O34 T203 O32B O36 O40 T215	O17 O32B O29 O32A T216B O47 T223 O34 T228 O2A O46 T203 O15 O4B T225B T235	T232 O17 O34 O12 T223 O4B O1A O32B O47 O40 O31 O15 T215	O31 O2A T232 O1A O17 O34 T215 O15 T223 O40	O34 O1B O4B T232 O17 O12 T223 O15 O47 O31 T215 O40	O34 O17 O2A O12 T232 O1A O47 O40 O4B T215 T223 O15	T223 O4B O32A O32B O31 O34 T232 O47 O15 O12 O40 T215	T203 T232 O26 O4A O8 O4B O17 T223 O32B O40 O47 O12 T215 O15 O31

Table 9.2: *The actual harvesting sequence for HF A to the left in the table is compared by period P to the eleven schedules generated throughout the 2009 milling season. Field O3B was not in the field records at the commencement of the season, and is therefore missing in the schedules. The DSSDV implementation was installed on a laptop computer which stopped functioning properly around 26 May, and required several days in order to be brought back to a functional state. After that, there were too few fields left to be able to achieve good solutions to the BMF (§6.1), which led to the decision to only use the SMF (§6.2) onwards.*

schedule is 50 %.

The second schedule (HF A BMF2) of the season is for the period 31 March 2009 onwards, its objective function value is within 4.1 % of optimality and the solution was uncovered after 2656 seconds, under a local search time limit of two hours. The fields T225 and T232 were scheduled for harvesting during the first month and field T232 was actually harvested approximately five months later. One reason for field T232 being harvested so late may be that it was not chemically ripened, unlike many of the other fields, which may have pushed it later in the real harvesting sequence. The SPD-1 value for this schedule is 50 %.

The third schedule (HF A BMF3) was generated under a local search time limit of thirty minutes, and the best solution was uncovered after 567 seconds, and its objective function value is within 1.78 % of optimality. The fields scheduled for harvesting during the first month are O31, T203, T235 and T225. Only T225 was actually harvested within two months. The SPD-1 value for this schedule is 25 %.

Schedule four (HF A BMF4) for HF A was generated on 25 April 2009 and the first period of the schedule began on April 27 2009. The local search time limit was two hours, the best solution was encountered after 5852 seconds and its objective function value is within 1.14 % of optimality. The fields scheduled for the first month of this schedule are T235, T225, O32A, T216 and T225B. Fields T225, T225B and T216 were actually harvested within one month from the schedule periods, but the other two fields were actually harvested three (T235) and four (O32A) months after the schedule periods. The SPD-1 value for this schedule is 60 %.

The fifth schedule (HF A BMF5) was generated upon receiving information concerning a number of events having taken place on each of the four farms. Five fields in HF A had lodged and six had been chemically ripened. The lodging was accounted for in the DSSDV by event-driven yield models and event-driven RV models similar to those described in Chapter 7. The input data were based on an eye-estimation performed by the growers of the extent of the lodging expressed as a percentage of each field having been affected. The time limit allocated for the local search solution procedure was thirty minutes and the best objective function value was uncovered after 208 seconds. The fields scheduled for the first month of this schedule are O36 and T232. The field O36 was actually harvested three months later than the scheduled date, while T232 was actually harvested five months later than its scheduled date. The schedule's objective function value was notably worse than previously, being within 27.1 % of optimality. It was concluded that this was a result of the combinatorial nature of the problem. More specifically, the problem was infeasible in terms of the minimum amount of cane to be harvested during each period, leading to some periods being allocated insufficient cane. The DSSDV solver uses penalties to encourage feasibility, and thus penalised these "infeasible" schedules. There were also problems of the opposite nature, where some periods were over-scheduled. In fact, some of the fields were moved to periods of their individual lowest yield, due to the decrease in penalty on the objective function value (penalty for exceeding the maximum allowed tonnage for the period during which the field was scheduled). Other problems also arose—due to the same basic problem—where fields were being combined based on tonnage so as to avoid breaking the minimum and maximum tonnage for each period, again putting profit aside for the benefit of reducing DSSDV solver penalties. An obvious attempt to remedy the situation is to increase the allowed maximum tonnage and decrease the allowed minimum tonnage for all periods, but that causes an imbalance in terms of consistent cane yield tonnage throughout the season. The SPD-1 value for the fifth schedule is 0 %.

Schedule six (HF A SMF6) was generated on 10 June 2009[2] using the SMF model incorporated into the DSSDV and the tabu search time limit was one hour. The fields scheduled for the first month of this schedule are O46, O29, O3C, T235, O32A, O19, O16, O13 and T234. The fields O13, O16, O19, O29, T234 and T235 were actually harvested within two weeks from their scheduled dates, O32A was actually harvested three months after its scheduled date, while O3C

[2]No schedule was generated for HF A for the period 26 May 2009 to 9 June 2009.

and O46 were actually harvested four and five months after their scheduled dates, respectively. From this schedule onward, the schedule printouts contained information on events having occurred, which for example, included information on which fields had been chemically ripened. The SPD-1 value for the sixth schedule is 67 %.

Schedule seven (HF A SMF7) was generated on 22 June 2009 and the first day of the schedule was 23 June 2009. It was generated under a tabu search time limit of eight hours. The fields scheduled for the first month of this schedule are O3C, O32B, O46, O23A, T203, T235, O19, O13 and T234. Fields O32A, T203, T235, O19 and O13 were actually harvested within two weeks from their respective scheduled dates, field T234 was actually harvested two months after its scheduled date, while fields O3C, O32B and O46 were actually harvested more than two months outside of their scheduled dates. For this schedule, the manager mentioned that the scheduling of ripened fields was correct. The SPD-1 value for this schedule is also 67 %.

Schedule eight (HF A SMF8) for HF A was generated on 10 July 2009 under a tabu search time limit of thirty minutes. The fields scheduled for the first month of this schedule are O46, T235, T203, O32B, O32A, O3C, O36 and T228. Fields T228, T235 and O36 were actually harvested within two weeks of their scheduled dates, while field O32A was actually harvested within two months of its scheduled date. There had been a severe frost [126] during the previous period, but unfortunately no data on the frosted fields or the level of frost was conveyed in time for incorporation into this scheduling run[3]. The SPD-1 values for this and the remaining HF A schedules are not computed since the number of fields remaining is low, which inflates the SPD-1 value substantially.

Schedule nine (HF A SMF9) was generated on 1 August 2009 under a tabu search time limit of thirty minutes, lacking in information regarding the impact of the frost on the four farms[4]. The schedule was, however, well received with a grade of 4.

Schedule ten (HF A SMF10) for HF A was generated on 10 August 2009 and there were seventeen fields left to schedule for harvesting. The tabu search time limit was thirty minutes and the schedule was afforded a grade of 4.

Schedule eleven (HF A SMF11)—the last schedule generated for HF A during 2009—was generated on 25 August 2010 and contained three more fields than HF A SMF10, since these had been added to the set of fields by the growers. This schedule was also given a grade of 4.

In summary of the 2009 HF A validation experiment, no schedule was given a “very poor” grade, one out of the eleven schedules (the first schedule) was given a “poor” grade, three schedules were given “acceptable” grades, four schedules were given “good” grades and four schedules were given “very good” grades.

9.1.2 Harvesting front B

HF B is situated a few kilometres north of HF A, the area under cane being approximately three hundred hectares distributed across seventy-nine fields ranging in size from 0.7 to 10.5 hectares. The varieties present in this harvesting front at the onset of the 2009 harvesting season were N12, N16, N23, N29, N35, N37, N40 and N41. Two growers belonged to HF B and their cane was delivered to Eston mill or (a small part of the cane) to Noodsberg mill; Noodsberg mill is

[3]This would of course have been a prime opportunity to test the DSSDV's capability in terms of scheduling frosted farms.

[4]This lack of information was due to the difficulties and resources required in assessing the level of frost damage sustained, not as a result of failing to convey the information.

							HF B DSSDV Schedules									
P	Actual		BMF1		BMF2		BMF3		BMF4		BMF5		BMF6		SMF6	
18/3–30/3	L238 L239 L183	C26 C28	C154 L7D L9D L40 L195	L236 L249 L21 L22	—		—		—		—		—		—	
31/3–14/4	L21 L22	C26 C28	L8 C150 L74 L84	L232 L264 C26 C28	L9D L40 L195	L249 L21 L22	—		—		—		—		—	
15/4–26/4	L7D L1D		C148 L70	L222	C154 L5D L12	L74 L84 L232	C154 L6 L23D	L40 L74 L249	—		—		—		—	
27/4–10/5	L9D L198A L200	C148 C147 C146	C29 C145	L36 L169	C146 L8D	L99 L222	C16 C24 L8D L9D	L84 L201 L219 L251	C146 L9D	L70 L326	—		—		—	
11/5–25/5	C153 C152 C151	C150 L70 L307	C147 L3D	L91	L9 C17 C24	C150 L70	C147 L70	L91	L8 C29 L74	L84 L166	C136 C145 L36	L40 L74 L249	—		—	
26/5–8/6	L70 L307 L174	L176 L12	C16 L6	L325	C16 C29 C138	L15D L264	L29D L176	L199 L242	L29D L32D	L36 L241	L9 C24 L14D	L84 L195 L232	L8 L9 C136 C145 L17D	L40 L84 L195 L219 L249	L219 C145 L164A	L250 L251 L195
9/6–22/6	L28D L236	L232 L222	C138 L8D	L23D L219	C136 C148	L181	C150 L7D	L100	L9 C136 C145 C148	L40 L249 L251 L307	L12 L41	L242 L251	L70 L164A	L232 L264	L70 L36	L249 L222
23/6–6/7	L169 L166 L181 L84	L249 L250 L29D	L7 L100	L251	C137 L200	L251	C148 L12	L14D L236	C24 C147 L31D	L198A L219 L264	C143 C144	L23D	L7 C143	L251	C143 L9 L8	L84 L7 C142
7/7–21/7	L23D L15D L14D	L219 L164A	L32D L200		C147 L35	L325	L7 L1D	L169 L198A	C142 C153	L169 L199	L7 L8	L100	C154 C153 C152 C151 C150 C148 C147	C146 L181 L28D L23D L14D L15D C24	L232 L300 L12 L40 C136	C24 L15D L14D L23D L28D
22/7–4/8	C17 C16 C154 C144	C143 C31 C32 C22	C146 L12	L176 L307	L8 L29D	L31D L91	C29 C145 C153	L5D L32D	C16 C154 L8D	L15D L28D L41	C17 L6	L35	L91 L166	L250	L181 C146 C147 C148 C150	C151 C152 C153 C154
5/8–19/8	L74 L100	C29 C11	C152 L41	L326	L23D L100	L303	C146 C152	L174 L307	C144 L3D	L35	L15D L112	L307	C142 L36	L222	L32D L169 L35	L264 L242
20/8–3/9	L200 L199 L242 L41 L40	L6 L91 L99 L112	C137 L35	L198A	C143 C151	L41	C138 L31D	L41 L232	C152 L181		L32D L91	L222	L3D L112		L8D L17D	L112 L41
4/9–30/11	L326 L325 L303 L300 L264 L251 L241 L201 L195 L36 L35 L32D L31D	L17D L8D L5D L3D C145 C142 C138 C137 C136 C24 L9 L8 L7	L14D L28D L199 C151 L15D L99 L9 C144 L5D L31D L17D L181 L242 L1D	L29D L166 L241 C17 C142 C153 L174 L201 C24 C136 C146 L112 L303	L14D L36 L169 L112 L166 L242 C144 C152 L28D L326 C142 L3D L6 L7	L174 L176 L201 C153 L1D L198A L199 L236 C145 L17D L32D L219 L241 L307	C137 L181 L35 L200 L8 C151 L15D L28D L36 C144 L99 L112 L9 C136	C143 L222 L241 L303 C17 C142 L17D L166 L264 L3D L195 L325 L326	L6 L12 L17D L174 C138 L195 L222 L242 L303 L5D L14D L23D L176 C17	C143 C150 L99 C151 L91 L112 L200 L201 L236 L325 L7 C137 L100 L232	L31D L166 L326 L70 L99 L174 L29D L169 C16 L3D L199	L303 C138 L5D L28D L201 L17D L176 L219 L236 L241 L264	L8D L32D L35 L6 L41 L325 L5D L100 L199 C144 L99 L176 L326 L12 L29D	L31D L169 C29 C137 L174 L201 L236 L303 C16 C17 C138 L241 L242 L300	L5D L201 L29D C144 C29 C16 L174 L91 L325 C17 L236 L241	L176 L6 L99 L326 L31D C137 L303 L100 C138 L3D L199

Table 9.3: *The actual harvesting sequence for HF B is compared to the first seven schedules generated throughout the 2009 milling season. These six first periods were scheduled using the BMF (§6.1) and the sixth period was also scheduled using the SMF (§6.2) for the first time.*

located approximately ninety kilometres to the north-east of the harvesting front and crushes approximately 1.4 million tonnes annually.

Tables 9.3 and 9.4 show all schedules that were generated by the DSSDV for HF B during the 2009 harvesting season.

The first schedule (HF B BMF1) was generated on 18 March 2009 under a one hour local search time limit and the objective function value of the best schedule uncovered was within 1.48 % of optimality. Of the seventeen fields scheduled for the first month, five fields (C26, C28, L21, L22, L7D) were actually harvested within one month after the scheduled period, two fields (L9D, C150) within two months, three fields (L236, L84, L232) within three months, one field (L249) within four months, two fields (C154, L74) within five months, one field (L40) within six months and three fields (L195, C8, L264) outside of six months. The SPD-1 value for this schedule is 41 %.

The second schedule (HF B BMF2) was uncovered in 2 268 seconds under a two hour local search time limit and the objective function value of the best schedule encountered is within 1.44 % of optimality. Of the twelve fields scheduled for the first month, three fields (L9D, L21, L22) were scheduled within one month from the actual harvesting period, two fields (L12, L232) within two months, two fields (L249, L84) within three months, two fields (C154, L74) within four months and one field (L5D) outside of four months. The SPD-1 value for this schedule is 42 %.

The third schedule (HF B BMF3) was uncovered in 5 512 seconds under a two hour local search time limit and its objective function value is within 1.35 % of optimality. Of the fourteen fields scheduled for the first month, only two (L9D, L84) were actually harvested within two months which yielded an SPD-1 value for this schedule of 14 %.

Schedule four (HF B BMF4) was uncovered in 5 538 seconds under a local search time limit of two hours, and its objective function value is within 1.2 % of optimality. Fields C146, L9D and L70 were actually harvested within two weeks from their scheduled date, fields L84 and L166 within two months, while fields L74, C29, L8 and L326 were actually harvested within five months from their respective scheduled dates. Twenty-one fields had lodged during the preceding period, and fourteen fields had been chemically ripened. The SPD-1 value for schedule four is 56 %.

The fifth schedule (HF B BMF5) was produced under an eighty minute local search time limit; the best objective function value was uncovered in 718 seconds and is within 1.47 % of optimality. Twelve fields were scheduled for the first month and fields L249, L84, L14D and L232 were actually harvested within two months from their respective scheduled dates. The combinatorial issues encountered while finding a schedule for HF A's fifth schedule were not noticed for HF B, other than some slight infeasibility in terms of over-allocation of tonnage during some of the periods. One may conclude that the problem was slight because of the closeness of the objective function value to the linear programming relaxation's optimum (1.47 %). The SPD-1 value of the schedule is 33 %.

The sixth set of schedules for HF B were generated on 27 May 2009 as one BMF-based schedule (HF B BMF6) and one SMF-based schedule (HF B SMF6). The SMF was operational two weeks earlier for HF B than for HF A, and the generation date for the sixth set of schedules was 26 May 2009. The local search time limit was two hours for the BMF and the tabu search time limit was four hours for the SMF. The BMF scheduled fourteen fields for harvesting during the first month, while the SMF scheduled ten for the same period. Both the BMF and the SMF scheduled fields C145, L70, L164A, L195, L219 and L249 for harvest during the first month and L17D is the only field which was scheduled more than two months apart by the two schedules. The SPD-1 value for the BMF schedule is 43 %, while the SPD-1 value for the SMF schedule is 60 %.

Schedule set seven (HF B BMF7 and HF B SMF7) was generated on 11 June 2009. The two schedules were obtained under two-hour and four-hour local/tabu search time limits, respectively. The best objective function value of the BMF schedule was 238 % from optimality, due to an infeasibility arising from an error in the input data with respect to the amounts of cane to be delivered during each period (D_j^{min} and D_j^{max}). The error in the input was very large

		HF B DSSDV Schedules									
P	Actual	BMF7	SMF7	BMF8	SMF8	BMF9	SMF9	BMF10	SMF10	BMF11	SMF11
9/6–22/6	L28D L236 L232 L222	C137 C138 L241 L250	L219 L74 C145 L195 L249 L164A	—	—	—	—	—	—	—	—
23/6–6/7	L169 L166 L181 L84 L249 L250 L29D	L31D L164A L199 L300 C24 L15D L14D L23D L28D	L36 C142 L40 L251 L250 L23D L15D	C9 C144 C145 L40 L74 L219 L249	L74 L195 L219 L251 C145 L250	—	—	—	—	—	—
7/7–21/7	L23D L15D L14D L219 L164A	L181 C146 C147 C148 C150 C151 C152 C153 C154	L14D L219 L164A L181 C146	C24 C154 L181 L23D L14D L15D	L164A C136 C9 L84 L36 L40 L249	L40 L219 C24 L15D L14D L23D C154	L164A L74 L219 L251 L40	—	—	—	—
22/7–4/8	C17 C16 C154 C144 C143 C31 C32 C22	C17 L166 L169 L219 L232 L236 L249	C147 C148 C150 C151 C152 C153 C154	L29D L164A L169 L250	C24 C154 L181 L23D L14D L15D	C143 L195 L251 L325	C24 L15D L14D L23D C154 C136 L36 L195	L32D L112 L251	L251 C145 L195 L74 C136 L40 C8	—	—
5/8–19/8	L74 L100 C29 C11	C8 C16 C136 C143 L84 L112	C136 L84 C143 C9	C7 C143 L35	C7 C8 C142 C143	C142 C144 C5D L41	C7 C145 L300 C8 C9 C143 L264	C8 C16 C6 L264 L325	L36 C9 L264 C142 L300 C143	C136 C145 C5D L35 L40 L195 L300	C145 L40 L195 L74 L251 C136 C8 C9
20/8–3/9	L200 L199 L242 L41 L40 C6 L91 L99 L112	L99 L201 L242 L264 L326	C7 C8 L232 L242 L32D L201 L264	C17 C136 C142 L84 L241 L251	L264 C6 L17D L32D L242	C7 C8 C145 C8D L32D	L201 C142 L242 L32D	C142 C143 L17D L40 L242	L32D L201 L242 C7 L17D	C8 L32D L112	L36 C142 C7 L264 L300
4/9–30/11	L326 L325 L303 L300 L264 L251 L241 L201 L195 L36 L35 L32D L31D L17D C8D C5D C3D C145 C142 C138 C137 C136 C24 C9 C8 C7	C6 L74 L222 C7 L41 L91 C144 C3D L29D C142 C145 L325 C5D L36 L251 L32D L35 L40 C9 C29 C8D L17D L195 L100 L303	L169 L35 C29 L112 L29D L300 C8D L222 L303 L17D L236 L41 L91 L166 C3D L326 C6 C144 C16 C5D L99 C17 L100 L31D L325 L241 C137 L199 C138	C8 C8D L41 L195 L201 L325 L32D L166 L326 L31D L36 L112 C137 C3D L99 C6 L17D L91 L300 C29 L100 L199 L264 C16 C138 L5D L242 L303	L35 L300 C29 C144 C8D L112 L169 L166 C5D L303 C17 L100 L91 L201 L41 L29D L325 L326 C16 L99 C3D L241 L199 C138 L31D C137	C9 L35 L112 L29D L74 L100 C16 L31D L91 L164A C138 L17D L242 L264 L303 C29 L36 L99 L201 C136 C6 L241 L326 C17 C137 L3D L199 L300	L112 L35 L29D L17D C8D L99 C29 C5D L41 C144 L100 L91 L325 L326 C6 C16 L31D L303 C17 C137 L199 C138 L241	C7 C145 L35 L41 L195 L300 L74 L91 L201 L326 C29 C137 L100 L241 C9 C17 L31D L36 L199 C138 C144 C5D C8D C136 L3D L99 L303	L112 L35 C8D L326 C6 C29 L91 L325 L41 C144 C17 L99 C3D C16 C5D L31D L199 L241 C137 L303 L100 C138	C9 C142 L36 L325 C29 L74 L99 L251 C7 C6 L201 L326 C3D L91 L100 C137 C138 L17D L199 C8D L31D L41 L241 L242 L264 L303	L112 L201 L32D L242 L17D L35 C29 C5D L41 C6 L91 C8D L303 L99 L100 L31D L326 C3D L325 C138 L199 L241

Table 9.4: *The actual harvesting sequence for HF B compared to the last ten schedules generated throughout the 2009 milling season. Note that for each period, one BMF harvesting schedule and one SMF harvesting schedule were generated by the DSSDV.*

and hence all solutions were penalised by the solver, leading to no apparent relative differences between different solutions with respect to objective function value. However, there may arise unknown such differences, so HF B BMF7 is not further analysed here. The SMF scheduled the fields L219, L74, C145, L195, L249, L164A, L36, C142, L40, L251, L250, L23D and L15D to be harvested during the first month. The SPD-1 value of HF B SMF7 schedule is 54 %.

Schedule set eight (HF B BMF8 and HF B SMF8) was generated on 23 June 2009 under local/tabu search time limits of thirty minutes and two hours, respectively. The best objective function value uncovered by means of the BMF was within 22 % of optimality, the problem instance again being infeasible, but not due to input errors. The infeasibility was due to the manner in which all chemically ripened fields had been represented, namely as a single field, which became slightly too large for any of the periods. The first month of the BMF contains the fields C9, C144, C145, L40, L74, L219, L249, C24, C154, L181, L23D, L14D and L15D and has an SPD-1 value of 77 %, but should be considered to be influenced by the fact that there was a small proportion of the season left at this stage. The first month of the SMF schedule contains the fields L74, L195, L219, L251, C145, L250, L164A, C136, C9, L84, L36, L40 and L249 and has an associated SPD-1 value of 54 %.

Schedule set nine (HF B BMF9 and HF B SMF9) was generated on 11 July 2009 with both the BMF and the SMF being subjected to local/tabu search time limits of one hour. The best objective function value uncovered by the BMF was within 2 % of optimality and has fields L40, L219, C24, L15D, L14D, L23D and C154 scheduled to be harvested during the first two weeks, while fields L219, L15D, L14D, L23D and C154 were actually harvested within one month from the scheduled dates. No SPD-1 value was computed due to the inflation arising from reaching the end of the season, but the BMF schedule is desirable judging by the five out of seven fields having been harvested within a month from the scheduled date. The SMF scheduled fields L164A, L74, L219, L251 and L40 to be harvested during the first two weeks, and fields L164A, L74 and L219 were actually harvested within one month from the scheduled dates. The SMF was also good, judging from the three out of five scheduled fields being harvested within one month.

Schedule set ten (HF B BMF10 and HF B SMF10) was generated on 1 August 2009 with a local search time limit for the BMF of thirty minutes and a tabu search time limit for the SMF of one hour. The best objective function value of the BMF is within 2 % of optimality. The BMF scheduled fields L32D, L112 and L251 to be harvested during the first two weeks, while field L112 was actually harvested within one month from the scheduled date. The SMF scheduled fields L251, C145, L195, L74, C136, L40 and C8 to be harvested during the first two weeks, while fields L74 and L40 were actually harvested within one month from the scheduled dates.

Schedule set eleven (HF B BMF11 and HF B SMF11) was generated on 10 August 2009 under local/tabu search time limits of two minutes and one hour for the BMF and SMF, respectively. The best objective function value obtained by BMF local search is within 1.3 % of optimality and the BMF scheduled fields C136, C145, C5D, L35, L40, L195 and L300 to be harvested during the first two weeks. Only one field (L40) was actually harvested within one month. The SMF scheduled fields C145, L40, L195, L74, L251, C136, C8 and C9 to be harvested during the first two weeks, and fields L40 and L74 were actually harvested within one month.

9.1.3 DSSDV appraisal by field experts

The harvesting front manager's comments and schedule evaluation grades as well as the knowledge gained in response to the feedback on all twelve sets of schedules generated throughout the season are presented in Table 9.5. During the first two weeks of the validation experiment described in §9.1.1 and §9.1.2, it became clear that chemical ripening must be taken into con-

	HF A		HF B		
P	Manager Comment	Grade	Manager Comment	Grade	Knowledge gained
18/3–30/3	N35 and N37 have relatively high RV % in immature cane. Current season sees relatively immature cane. Not practical to move equipment between farms too often.	2	All harvested fields were ripened. Manager thought L21 and L22 were good, but weather interfered.	3	Ripening must be incorporated into all scheduling.
31/3–14/4	Cane is immature.	5	Ripened fields go first.	3	
15/4–26/4		5	Fields are destined for plough-out.	3	Plough-out decision must force scheduling before end of June.
27/4–10/5		5	Again, fields are destined for plough-out.	3	
11/5–25/5	Good.	5	You have not taken plough-out fields into consideration yet. Your schedule will not be accurate.	3	
26/5–8/6	Age computations may be wrong.	4	We cut L307 because we needed an infield for the weekend.	3	
9/6–21/6		3	The fields cut were convenient (positionally) on farm.	3	Distance matrix may be necessary.
23/6–6/7	Cutting plough-out and ripened fields.	3	Good. Cutting plough-out and ripened fields.	4	
7/7–21/7	Still cutting plough-out and ripened fields.	3	Cutting ripened fields.	3	
22/7–4/8	Harvested fields with name starting with a “T” were all frosted.	4	Fields whose names end with a “D” were frosted. L219 and L164 are good choices.	4	Frost definitely must be part of any DSS for sugarcane harvest scheduling.
5/8–19/8	The harvested fields were all ripened.	4	The frosted cane was unpredictable and the other [harvested] cane was ripened.	4	
20/8–3/9	Harvested fields were all ripened and in the schedules. Good.	4	The frost cane was unpredictable. There are actually no wrong fields to harvest now. Most cane is ready. We [the growers and the manager] are choosing fields with rain and position on farm in mind.	3	

Table 9.5: *The appraisal forms contained comments and grading by the manager, and conclusions in the form of knowledge gained were drawn after every period. The grades are on a scale of 1 to 5, where 1 is very poor, 2 is poor, 3 is acceptable, 4 is good and 5 is very good.*

sideration during scheduling decisions. It also became clear that some varieties are known to perform better than others under cold and wet conditions, such as N35 and N37.

Later during the validation, it became clear that the participating growers tend to decide early in the season which fields to plough out, and that this information was not correctly captured ahead of the onset of the 2009 harvesting season. The capturing and handling of plough-out and chemical ripening related decisions became a prioritised improvement to be implemented in preparation of the follow-up 2010 validation experiment.

The growers took travelling between fields into consideration, and it was decided to incorporate this into a formulation presented in Chapter 11, the future work chapter presented later in this dissertation.

Further into the harvesting season (during July) a frost struck a significant number of fields across both harvesting fronts. Information about the extent of the frost with respect to severity and area affected never became available.

Towards the end of the 2009 harvesting season, the author attended the growers' end-of-season meeting. Some final remarks on the DSSDV were collected and it was agreed to continue the validation experiment and development project during the first quarter of the 2010 harvesting season.

Some statements made by the growers during a final discussion at the end-of-season meeting include the following:

- The growers felt that the DSSDV may be useful as a guide, not in detail.
- They thought the DSSDV may be more useful to large estates (in fact, that it will only become useful for estates producing more than 25 000 tonnes per annum).
- The DSSDV sometimes schedules fields wrongly, which may cause losses if the responsible person is not careful.
- It is only necessary to update schedules once per month.
- Some flat fields should be scheduled for every month, and
- the DSSDV must take dry conditions into consideration.

After this validation experiment, the next step was to completely rebuild and solidify the DSSDV into a new implementation of the DSS architecture that could handle all requirements and other problems put forward during 2009. These included coding the DSS implementation in a different programming environment accessible by the manager of a typical harvesting front, refitting all the regression models and updating the handling of certain events and decisions that are not automated within the DSSDV. This was not possible during the process of validating the DSSDV. This was done during the break between the 2009 and 2010 harvesting seasons.

9.2 Validation of the final DSS

The second stage in validating the conceptual DSS proposed in this dissertation consisted of employing the computer implementation of the DSS of §8.2. Two schedules were generated for each harvesting front described in §9.1.1 and §9.1.2, a set consisting of one schedule for each harvesting front generated in March 2010 and another set of one schedule for each harvesting front generated in May 2010.

9.2.1 Hindsight schedules of the 2009 harvesting season

The 2009 harvesting season field database was input into the newly developed DSS, which was not available until the end of that season. The field conditions at the onset of the 2009 harvesting season were captured into the DSS databases, prediction models were employed to compute the parameters required by the scheduling model and three schedules were obtained. Fields that were newly added to the set of harvesting fields during the season were not included in the schedules, since no information is available regarding the status of those fields at the onset of the season. Fields in the actual harvesting sequence that were actually harvested over several time periods of the schedule were here assumed to have been completely harvested at the first occasion. The split field O4AB was considered as a whole again, and was assumed to have been completely harvested at the time of the harvesting of field O4A. The two first schedules are shown in Table 9.6 together with the actual harvesting schedule of the two harvesting fronts. The first two schedules are hindsight schedules for HF A and HF B, while the third schedule was generated by the treating the two harvesting fronts as a single harvesting front.

The total profit computed by the DSS for the first two best hindsight schedules combined is R8 369 729. The best solution found when solving for the third hindsight schedule has an objective function value of R8 395 032, which is R25 303 more than the two first schedules combined. This indicates that combining two harvesting fronts into one may yield an increased profit solely due to the different conditions under which the fields may be sequenced. Any advantages in terms of economy-of-scale are in addition to the advantage in sequencing conditions.

The aim of the computer implementation of the DSS is to enable the testing of the DSS as a concept with respect to its ability to provide support in making harvest scheduling decisions. In §9.1, this question was explored by means of appraisals from industry professionals. Another way to explore the question is to examine the schedules statistically. The measure used for this task first takes the difference in the number of days between the day on which a field was actually harvested compared to when a hindsight schedule listed the field for harvesting. The absolute values of these differences are summed for each field, and then divided by the number of fields; the measure is thus the average difference in days between field harvesting date and schedule date. This value is denoted by δ^{hs}. The null hypothesis is that the DSS schedules are not better predictors of harvesting sequences than a randomly generated schedule, *i.e.*

$$\begin{aligned} \text{H}_0: &\quad \delta^{\text{hs}}_{\text{DSS}} \geq \delta^{\text{hs}}_{\text{RND}}, \\ \text{H}_1: &\quad \delta^{\text{hs}}_{\text{DSS}} < \delta^{\text{hs}}_{\text{RND}}, \end{aligned}$$

where $\delta^{\text{hs}}_{\text{DSS}}$ is the value of δ^{hs} for schedules generated by the DSS computer implementation and $\delta^{\text{hs}}_{\text{RND}}$ is the δ^{hs}-value for schedules generated randomly. To test the hypothesis for HF A, thirty-five schedules were randomly generated. There is only a single estimate available for $\delta^{\text{hs}}_{\text{DSS}} = 58$, so the appropriate test is a one-sided Student's t-test. First, the probability of sampling a value $X \leq \delta^{\text{hs}}_{\text{DSS}}$ from the randomly generated schedules' $\delta^{\text{hs}}_{\text{RND}}$-distribution was computed. Assuming that

$$\frac{\delta^{\text{hs}}_{\text{RND}} - \bar{\delta}^{\text{hs}}_{\text{RND}}}{s_{\delta^{\text{hs}}_{\text{RND}}}}$$

is distributed according to a standardised Student's t-distribution, the probability of sampling a value X less than 58 is 0.013, which indicates that the null hypothesis is not true ($\bar{\delta}^{\text{hs}}_{\text{RND}} = 80.9$ and $s_{\delta^{\text{hs}}_{\text{RND}}} = 9.77$). The corresponding test for HF B yielded a p-value of 0.048, also indicating that the schedules generated by the DSS are better predictors of harvesting sequences than randomly generated schedules[5], as expected.

[5]The probability of the statement "*The DSS schedules are better than randomly generated schedules*" being

HF A Actual	HF A Hindsight	Difference in Harvesting day	HF B Actual	HF B Hindsight	Difference in Harvesting day	HF B Actual	HF B Hindsight	Difference in Harvesting day
O23	T228	−19	L238	L008D	−75	C016	C153	−40
O26	T225B	−21	L239	L074	−69	C154	L181	31
O4AB	T216B	−205	L183	L200	−47	C144	C150	−72
O8	O46	−59	C026	L084	−71	C143	L005D	96
T225	O23	−104	C028	C136	−71	L074	C138	139
T216	O26	−28	L021	L300	−26	L100	L264	−38
O17	O42	5	L022	L241	−25	C029	C017	−47
O42	O36	41	L007D	L169	−210	L199	L303	−3
O41	O29	−88	L001D	L166	−179	L242	L176	13
T231	O31	−10	L009D	L307	−199	L041	L242	−61
T225B	T235	80	L198A	L249	−143	L040	L236	−62
O16	O17	11	L200	L070	26	L006	L326	−22
O29	O8	47	C148	C143	−52	L091	L174	−41
O19	O16	0	C147	C009	−75	L099	L031D	−33
O13	T216	−1	C146	L021	−61	L112	L199	119
T228	T231	114	C153	L022	−74	L326	C016	28
T235	O19	65	C152	L232	−63	L325	L012	−16
T216B	O13	115	C151	L164A	−56	L303	L003D	43
O1A	O32A	−16	C150	L250	−75	L300	L198A	171
O1B	T225	−32	L070	L183	25	L264	L201	52
O2A	T232	−54	L307	L112	32	L251	L017D	−49
O36	O3C	109	L174	C137	−88	L241	L029D	178
T223	O1A	−28	L176	C007	−74	L201	L100	22
T215	T203	−67	L012	L219	−93	L195	L006	−44
O40	O41	−27	L028D	C142	−4	L036	C029	−32
T203	O1B	−1	L236	L239	−70	L035	L325	−21
O47	T223	−68	L232	C008	33	L032D	C144	−9
O32A	O40	50	L222	L238	−132	L031D	L001D	55
O31	O2A	120	L169	L028D	68	L017D	L099	37
O12	O32B	−66	L166	C026	67	L008D	L091	217
O15	O15	−23	L181	C028	−28	L005D	L222	90
O32B	O4AB	−11	L084	C148	81	L003D	L032D	54
O34	T215	−40	L249	L023D	68	C145	L041	−14
O3C	O47	65	L250	C154	47	C142	L009D	159
O46	O12	203	L029D	C024	−76	C138	L040	98
T232	O34	99	L023D	L015D	15	C137	L035	172
			L015D	L014D	12	C136	L007D	223
			L014D	C146	13	C024	L036	145
			L219	C151	46	C009	C145	203
			L164A	C152	68	C008	L251	170
			C017	C147	−20	C007	L195	183

Table 9.6: *Actual harvesting sequences for HF A and HF B compared to hindsight schedules for these fronts suggested by the final DSS. Here* "HF A Actual" *means the actual harvesting sequence which reads from top to bottom.* "HF A hindsight" *is the hindsight schedule generated after completion of the season. The column* "Difference in harvesting day" *shows the difference in harvesting day for the field in the* "Actual" *column, i.e. the number of days that elapsed between its actual and hindsight-scheduled harvesting days (actual day minus hindsight day). The corresponding meanings are valid for the HF B columns. The second set of columns for HF B contains the second halves of the HF B schedules.*

9.2.2 Harvesting front A

The first schedule for HF A (HFA1) was generated on 16 March 2010 under a tabu search time limit of five minutes, and the best objective function value found was within 2.5 % of optimality. The entire final harvesting sequence is not known, so neither SPD-1 nor SPD-2 could be employed as measures of prediction desirability. Instead, another measure was defined as the average difference between the actual harvest day and the scheduled harvest day of the first ten (or less than ten if ten are not available) harvested fields, this measure being called SPD-3. This schedule and the other three schedules generated during the 2010 validation experiment are shown in Table 9.7 together with the actual harvesting sequences. The SPD-3 value (based on ten fields) for the first schedule is ninety days.

The second schedule for HF A (HFA2) was generated on 10 May 2010 under a tabu search time limit of two minutes, and the best objective function value found is within 5.6 % of optimality. The SPD-3 value (based on three fields) for the second schedule is sixty-one days.

wrong is $1 - (1 - 0.013)(1 - 0.048) = 0.06$, if the combined probability is considered.

		HF A			HF B				HF A		HF B	
Day	Actual	HFA1	HFA2	Actual	HFB1	HFB2	‖	Day	HFA1	HFA2	HFB1	HFB2
77	O18	O3A		C12	L173			152			L90	C3
79				C13	L1			154	T224		L327	L83
80				C14	L235			156			L171	L302
82	O20	O25		C18	L14			160		O6	C20	C4
83				C19	L7			162	T226	O42B	C24	L224
85				C20	L177			165		T227A	L306	
86	O28	O7		C21				167		T227	L76	L185
88				C24				169			L57	C10
89				C25	A2			173			L34	C15
91	O32B	O24		C135	L234			175		O13	L172	L78
92				C145	A25			177	O39		A4	A25
94				C149				179			L56	C1
95	O44	O22		A2				181	O43	O7	L197B	L168
97				A6				183	O9	O22		L221
98				A10	L223			185			L62	L82
100				L7				188		O27	L168	L14
101	O45	O27		L34				190	O2B	O10	L251	L90
103				L56				192	T216A		C141	L108
104				L57	L11			194	O45		A10	L223
106	T216	O10		L76	L324			196	T227	O35	L003	L177
107				L77	L108			200	T236		L197A	L179
109				L229				202	O28	O3D	A21	L251
110	T216B	T232		L1				204	O32B		C145	L324
112				L8	L221			206			C135	L234
113				L10				208	T216	O24	L240	
115	T224			L11				211	O44	T226	L224	C141
116		O3D		L186	L78			213	O13		L264	
118				L231	L10			217			C1	C140
119	T232			L264	C18			219			C2	C6
121		O18		L266	L229			221			C149	
122				L327	C25			223	T227A		L23	L172
124	T236		O25	L328	C14			225			C3	
126			O9	C4	C13	L173		229			L303B	L3
128				C3	L43	C2		231	O23		L083	
130		O20		L177	L8	L240		234		O43	C4	
132	O13			L43	C12	L198B		236	O6		C6	C5
134				L83	L186	L43		238		O14	C10	A4
136				L34	L77	L23		240			L185	C23
138	O14			L229	A6	A21		244	O42B		L220	L2
139			T216A	L220	C140			246	O21	O21	C23	L303B
141				L327	C21	L197A		248			L302	L171
143		O35	O3A	L328	L179			250			C5	
145	O37			L164	L82	L235		252			L2	
147				L186	C19			254	O14		A15	L197B
149				L251	L266			257		O26		L220
151			O37	L172	L90	C3		259			L198B	L308
								261			L164B	
								263		O39	C15	A14
								265	O37		L308	
								267			A24	
								269			L231	
								273			A14	L306
								275		T233	L195	L164B
								277			L300	A15
								279	O26		L36	A24
								282	O38	O23	L219	C27
								284	T234		L22	L96
								286	T233	T234		L219
								288		O5A	C27	L36
								290			L328	L22
								292	O30	O2B	L96	L195
								294	O5A	O38		L300
								298	O5B	O5B		
								300		O30		

Table 9.7: *Actual harvesting sequences for HF A and HF B until end of May 2010 compared to the four schedules HFA1, HFA2, HFB1 and HFB2, shown together with an approximate day of the year. The schedules extended until early 2010, which is shown in the rightmost part of the table.*

9.2.3 Harvesting front B

The first schedule for HF B (HFB1) was generated on 16 March 2010 under a tabu search time limit of ten minutes, and the best objective function value found is within 6.5% of optimality. The SPD-3 value (based on ten fields) for the first schedule is sixty-two days.

The second schedule for HF B (HFB2) was generated on 10 May 2010 under a tabu search time limit of ten minutes, and the best objective function value found is within 9.5% of optimality. The SPD-3 value (based on 8 fields) for the second schedule is fifty-nine days.

9.2.4 DSS appraisal by field experts

Each of the four schedules displays a rather large SPD-3 value when compared to the best SPD-2 values of Table 9.1. The cause for this is probably a lack of communication between the author and the harvesting front manager, since it later came to the author's attention that conditions at the four farms were dry. The dry conditions had caused the growers to schedule high-lying fields earlier.

Two additional schedules were generated in hindsight, taking into consideration that conditions were dry. All fields had the event called "Drought stress" set to 100 % of the area, the event date being 17 March 2010. The first hindsight schedule is for HF A and it achieved an SPD-3 value of fifty-eight days, an improvement on the ninety days of schedule HFA1 of thirty-two days. The second hindsight schedule is for HF B and achieved an SPD-3 value of sixty-two days, which was one day worse than that of HFB1.

The final comments on the DSS were communicated personally, and the manager of the harvesting front said that if he were less experienced and lacked the support of the growers he would have liked to have the DSS generated schedules as support. He also felt that the schedules that had been sent for the 2010 season were very good [126].

9.3 Chapter summary

In this chapter, a DSSDV validation experiment was described, in which a number of schedules were sent to the manager of two harvesting fronts of a syndicate comprising four growers in the Eston Mill area of KwaZulu-Natal. The growers and the manager were available during the entire 2009 harvesting season and provided appraisals and other feedback on each schedule. In general, the schedules were appraised by the manager and growers as acceptable, good or very good. The growers are of the opinion that the schedules may be of particular assistance when scheduling larger estates.

The DSSDV was further validated by computing two numerical measures providing a scoring method with respect to the ability of the DSSDV to predict the actual harvesting sequence. The first measure, SPD-1, showed that on nine out of fifteen occasions, the DSSDV scheduled the fields in such a way that more than half of them were actually harvested within two months of the scheduled date. The second measure, SPD-2, showed that there was no significant difference between the BMF-based and SMF-based schedules in their ability to predict the harvesting sequence, but it was mentioned that the BMF-based schedules were harder to generate for certain problem instances. The SPD-2 values did, however, show that there is a significant difference in prediction desirability attributable to HF A than to HF B, but the reason for this is unknown. Overall, the SPD-2 values indicate that the DSS on average scheduled fields approximately sixty-five days from their actual harvest day for the particular experiment in question. The SPD-2 values of the six SMF-based schedules for HF A were, however, in the range twenty-nine to fifty-one, indicating that prediction desirability may be better under certain conditions.

This chapter also contains a description of the effort undertaken to validate the DSS. The DSS as a computer implementation is based on the same conceptual decision support system architecture that formed the basis of the DSSDV, and hence the DSSDV validation is also applicable to the DSS. The DSS was tested by means of hindsight scheduling the 2009 harvesting season and was further validated during the early part of the 2010 harvesting season by means of two sets of shadow-schedules of the two harvesting fronts HF A and HF B. It was necessary to introduce a third performance measure, called SPD-3, which indicated that the average difference in days between the harvested fields' actual harvesting dates and their DSS scheduled harvesting dates was approximately sixty days, and in one case ninety days. Two additional hindsight schedules were generated thereafter, taking into consideration recent information about dry conditions prevailing across the farms. A result of comparing these hindsight schedules with dry conditions taken into consideration in addition to the original schedules was that the ninety day SPD-3 value for HFA1 was reduced to fifty-eight days. As a final remark, the harvesting front manager said that he would have liked to have the scheduling capability of the DSS as support had he been less experienced, and added that the schedules for the 2010 harvesting season were "perfect."

This chapter is in fulfilment of Dissertation Objectives V and VIII of §1.3.

Part III

Conclusion

CHAPTER 10

Conclusion

Contents

This chapter contains a summary of work contained in this dissertation, a presentation of the main contributions of this dissertation to the field of sugarcane harvesting decision support and an appraisal of these contributions.

10.1 Dissertation summary

Chapter 1 of this dissertation opened with a brief history of sugarcane cultivation in §1.1 where it was described how sugarcane followed the human migration routes from Melanesia and South-East Asia to India and through the campaigns of warring emperors reached the Middle East, the Mediterranean and northern Africa, was then carried by colonialist missions to the Americas, Mauritius and Reunion and finally taken by a Mr Morewood to South Africa in 1846. This was followed by a description of the current state of the South African sugarcane industry and an overview of some of the issues involved in producing sugarcane. Then the problem of providing decision support to managers charged with the task of scheduling sugarcane harvesting operations was introduced in §1.2. The aim of this dissertation was alluded to in a statement that the proposed solution to the decision problem considered is the schedule output by a DSS to be presented later in this dissertation. It was mentioned that the focus of the work throughout the dissertation would remain on medium-scale commercial growers. In §1.3, the nine objectives of the dissertation were presented. Chapter 1 was concluded by a delimitation of the scope of the dissertation in §1.4, in which several of terms used throughout the dissertation were defined, as well as a description of the organisation of material included in the dissertation in §1.5.

Chapter 2 opened in §2.2 with a review of the various approaches taken in the literature with respect to modelling sugarcane production in general. The modelling work by Grunow *et al.* [81], Higgins [96], Higgins *et al.* [101], Muchow *et al.* [158] and Piewthongngam *et al.* [170] exemplified the variety of contexts having been considered in literature as well as the complexities of implementing the models and gaining industry acceptance for those implementations. Several mixed integer linear programming models as well as discrete event simulation models appear in the literature, and the problems modelled range from highly delimited case studies to very broadly defined scenario-type exploratory questions. In Australia, quite a few models have reached the stage of providing actual decision support, and some of them are integer programming-based, while some are simulation-based. There are many models to be found in literature, but few OR tools of validated quality available to growers for seasonal harvest scheduling.

In §2.3, a survey of current research topics within the field of supply chain management was described, followed in §2.4 by a brief review of current agricultural value chain research with respect to sugarcane. Within the field of SCM-related research, theories are still under development. The scientific community does not yet agree on exactly how modelling in terms of, for example, performance measurement of supply chains should be undertaken in order to provide successful SCM-improvement projects. This dissertation may be called an SCM-related research project since it concerns "*...developing solutions or answers to specific supply chain-related problems or challenges...*"—a defining trait of SCM-research, according to Stock [208, p. 147]. In §2.4, it was argued that work done towards providing decision support in the sugarcane industry is more likely to be successful if the researcher performing the work does so in close collaboration with the potential user of the decision support [103]. It was also noted that scheduling models have been received with positive anticipation by the Australian sugar industry [203].

The various modelling approaches taken in the literature towards predicting the recoverable sugar and the cane yield of sugarcane was explored in §2.5. Simulation-based models such as APSIM-Sugarcane, CANEGRO or QCANE, are accurate under ideal cultivation conditions, but would have required extensive tuning in order to model recoverable value and cane yield in practical settings for individual farms, and the model developed for South African conditions (CANEGRO) could not compute RV, only recoverable sugar and cane yield. The simulation-based models were, furthermore, based on architectures too involved to "copy-implement" into the DSS of §8.2, and one of the objectives of this dissertation was to implement the DSS as a standalone system, effectively ruling out the use of the simulation-based models in this work. A viable option for predicting recoverable value and cane yield was found in the work by Greenland [79], Higgins [94], Higgins *et al.* [101], Jiao *et al.* [116], and Lawes and Lawn [132] who all used regression-based models to predict cane yield and recoverable sugar.

In §2.6, studies in the literature of various extraneous events such as harvest, fires of various kinds under different circumstances, frost of varying degree, lodging, flowering and Eldana infestation were examined in search of quantified resulting rates of deterioration in cane yield and recoverable value. Very few deterioration rates were found, most results directly usable in computing the parameter values of §7.4.2, §7.4.4 and §7.5.1 having been performed on fires (in [38, 234–236]), where deterioration rates were at the centre of the study. These papers set good examples of what should be done for the other events. The extraneous events were given further attention in §2.7 by analysing the information sheets published by SASRI. Commercial sugarcane varieties were tabulated according to a wide spectrum of properties and the common diseases of brown rust, mosaic, pineapple disease, pokka boeng, ratoon stunting disease, red rot, smut and sour rot were all discussed in terms of their effects on sugarcane harvest scheduling. The SASRI information sheets also render advice with respect to accidental fire, drought, frost and flowering, much apparently being drawn from the same work as that examined in §2.6.1, §2.6.2, §2.6.3 and §2.6.4. A description of the advice by SASRI on insect infestations by Eldana, Sesamia, Chilo, Hysteroneura and green leaf sucker concluded Chapter 2, which was presented in partial fulfilment of Dissertation Objectives I, III and IV, as described in §1.3.

Chapter 3 contains an overview of global sugar production (§3.1) followed by a series of brief overviews of the national sugarcane production scene, the rules and regulations governing the national sugar industry and a differentiation of growers according to size, for the South African case, in §3.2. In §3.3, the cultivation of sugarcane was briefly touched upon, and some examples of activities that growers perform and problems that they face on a daily basis were mentioned. This was followed by an overview description of the harvesting operation (§3.4) and the loading and transportation of sugarcane to the mill (§3.5). Chapter 3 was concluded by a brief mention of the various constituents of sugarcane and a selection of current uses of these constituents. Chapter 3 stands in partial fulfilment of Dissertation Objectives I, III and IV, and together with Chapter 2 achieve the fulfilment of these objectives.

In Chapter 4, the problem considered in this dissertation, called the *tactical sugarcane harvest scheduling problem* (THSP), was described in some detail. The problem description was structured according to various different viewpoints or contexts. There were six specific contexts: *small-scale solitary growers*, *small-scale harvesting groups*, *medium-scale commercial solitary growers*, *medium-scale commercial harvesting groups*, *independent large-scale commercial estates* and *mill-owned large-scale commercial estates.* The *decision support system design and development* approach taken towards solving the THSP for the medium-scale commercial solitary grower and medium-scale commercial harvesting group contexts was singled out for focus in the remainder of the dissertation and the main components of the validation process of the DSS were envisaged. The thesis of this disseration was stated in §4.4.

The CAP (§5.1.1), GAP (§5.1.2), ATSP (§5.1.3), VRP (§5.1.4), TDTSP (§5.1.5), ATSPTW (§5.1.6), ATSPTDC (§5.1.7), CJSP (§5.1.8) and STDSP (§5.1.9) were briefly reviewed in Chapter 5. These well-documented operations research problems were conceptually connected to the THSP. The ATSPTDC was formulated as an integer programming problem, such a formulation not having been encountered during the literature review of this dissertation. These well-known problems of §5.1 were described in partial fulfilment of Dissertation Objective II as stated in §1.3. In §5.2, the branch-and-bound method (§5.2.1) was described in general and in terms of the TSP. The cutting plane method was described in §5.2.2, also in general and then specifically for the TSP. These two exact methods are considered the starting points for most exact approaches towards solving *NP-hard* combinatorial optimisation problems. The work of §5.2 was in partial fulfilment of Dissertation Objective II as stated in §1.3. In §5.3, a number of popular metaheuristics, including ant colony optimisation, genetic algorithms, memetic algorithms, scatter search, simulated annealing and tabu search, were briefly reviewed and pseudo-code algorithms for these methods were given. These descriptions were in partial fulfilment of Dissertation Objective II as stated in §1.3. A number of practical aspects of the various solution methodologies were discussed in §5.4, completing fulfilment of Dissertation Objective II as stated in §1.3. The choice was made to base the core of the DSS on an alternative formulation of the ATSPTDC to the one presented in §5.1.7, the alternative formulation subsequently presented in §6.2.1, coupled with a tabu search metaheuristic subsequently presented in §6.2.2.

Two distinct decision support systems were developed in this dissertation, called the *decision support system—development version* (DSSDV) and *decision support system* (DSS), respectively. The optimisation models destined for the scheduling model building blocks of the DSSDV and the final DSS were formulated in Chapter 6. The scheduling model of the DSSDV, called the BMF, was described in §6.1 and approaches towards its solution were attempted by means of the off-the-shelf optimisation software suite (based on the branch-and-bound method) LINGO 9.0, the attempt being described in §6.1.1, as well as by means of a specially designed local search-based approximate solution algorithm described in §6.1.2. The core model of the final DSS, called the SMF, was solved using an attribute-based tabu search method incorporating a shift neighbourhood and an ejection chain, described in §6.2.2. Failed attempts, described in §6.2.1, at solving small instances of the SMF using LINGO 11 shows that it is a very hard problem. Even the LP relaxation could not be solved for practical size instances since the LINGO solver ran out of memory (4 Gb of RAM was installed on the PC involved). Chapter 6 stands in partial fulfilment of Dissertation Objectives VI and VII of §1.3.

Chapter 7 contains descriptions of the various models later employed within the prediction models building block of §8.1.2. A definition of the value of a sugarcane crop was introduced in §7.2 and was followed in §7.3 by a brief methodological review of multiple linear regression, the technique chosen as the basis for developing the models populating the prediction models building block. This was followed in §7.4 by a description of the *base yield models* of §7.4.1, developed using this technique. A manner in which to adjust the base yield models for practical circumstances under which certain extraneous events were assumed to have taken place was described in §7.4.2 by means of the so-called *event-driven yield models*, whose parameters were based on the findings described in Chapters 2 and 3. The *base RV models* of §7.4.3 were also developed using multiple linear regression and since RV is affected by extraneous events, these base RV models were also equipped with adjusting *event-driven RV models* populated with parameter values as a result of the findings in Chapters 2 and 3.

The cost parameter values of Chapter 7 assigned to the harvesting of a particular field at a particular point in time served the dual purpose of representing the actual cost of harvesting a particular field at that particular point in time (which, for example, varies due to the probability of rain) and representing an artificial penalty assigned to harvesting a field outside a time-window associated with certain extraneous events having occurred on the particular field. The artificial penalties were described in §7.5.1 and the more natural cost models were put forth in §7.5.2 and §7.5.3. Chapter 7 stands in partial fulfilment of Dissertation Objective VI of §1.3.

In Chapter 8, the architectural framework of the decision support system was presented (§8.1.1) and various building blocks were incorporated (§8.1.2) into it in order to formulate an approach towards solving the THSP for medium-scale solitary commercial growers and medium-scale commercial harvesting groups. The DSS architectural framework was implemented on a personal computer in a programming language available to those with access to Microsoft Excel 2007, as described in §8.2. Screen-shots and bulleted descriptions showed and described the appearance and workings of the various parts of the user interface and the procedures embedded within. The DSS computer implementation was verified in §8.3 by means of selected problem instances with designed parameter settings (§8.3.1, §8.3.2 and §8.3.3) and the results were discussed at some length, with conclusions drawn regarding the setting of the scheduling model parameters. Chapter 8, together with Chapters 6 and 7 completed the fulfilment of Dissertation Objectives VI and VII of §1.3.

Chapter 9 contains a description of the validation experiment performed on the DSSDV, in which a number of schedules sent to the manager of two harvesting fronts, HF A and HF B, of a syndicate comprising four growers in the Eston Mill area of KwaZulu-Natal were evaluated and used for the purpose of generating response in terms of faults, weaknesses, strengths or missing properties of the DSSDV schedules. The growers involved in this case study—also referred to as the validation experiment—and the manager were available during the entire 2009 harvesting season, and the schedules were in general evaluated by them as acceptable, good or very good (§9.1.1, §9.1.2 and §9.1.3). The DSSDV was also validated by means of a scoring method designed to measure the DSSDV's performance in terms of predicting the actual harvesting sequence. The first measure, SPD-1, indicated that most of the time (nine out of fifteen schedules) the DSSDV scheduled the fields within two months of their actual harvesting date (§9.1.1 and §9.1.2). The second measure, SPD-2, showed that the BMF-based and the SMF-based schedules did not yield a significant difference in terms of their ability to predict the harvesting sequence. According to the SPD-2 values, there is, however, a significant difference between HF A and HF B in prediction desirability (§9.1), the reason for which was not determined. The DSSDV on average scheduled fields approximately sixty-five days from their actual harvest day for the particular experiment in question, according to the analysis performed on the SPD-2 values. SPD-2 values in the range of twenty-nine to fifty-one for six SMF schedules for HF A shown in Table 9.1 indicated that predictions may be better under certain conditions. The DSS as a computer implementation was also validated, this experiment being described in §9.2. The DSS was tested by hindsight scheduling the 2009 harvesting season, the results of which are described in §9.2.1. Evidence was found that the hindsight schedules were significantly better at predicting the actual harvesting sequence than were randomly generated schedules, as expected. The DSS was further validated during the early part of the 2010 harvesting season by shadow scheduling the two harvesting fronts HF A and HF B, described in §9.2.2 and §9.2.3. A third performance measure—SPD-3—indicated that the average difference in days between the harvested fields' actual harvesting dates and their DSS scheduled harvesting dates was approximately sixty days, but for the first HF A schedule it was ninety days. During the grower and manager appraisal process, described in §9.2.4, it came to the author's attention that there had been a slight, unreported drought at the onset of the season, and these dry conditions had affected the actual harvesting sequence. Two hindsight schedules were subsequently generated, taking this belated information into consideration. These hindsight schedules were compared to the original schedules which lead to the interesting result that the ninety day SPD-3 value for HFA1 was reduced to fifty-eight days. During the appraisal conversations, summarised in §9.2.4, a field expert stated that the schedules would have been useful to him if he had been less experienced. Chapter 9 stands in fulfilment of Dissertation Objectives V and VIII of §1.3.

10.2 Main contributions of this dissertation

The contributions to the sugarcane harvest scheduling decision support problem made in this dissertation are outlined in this section. These contributions are described in the order in which they appear in this dissertation.

Contribution 1 *A contextually formulated description of the* tactical sugarcane harvest scheduling problem *in Chapter 4.*

The first contribution of this dissertation is the formulation of the THSP within several contexts. This problem formulation served as the object of the design of the decision support system architecture and has not been contextually defined previously in the literature.

Contribution 2 *An exploration of well-known mathematical programming models and their relationships with the THSP within its various contexts in §5.1.*

In §5.1, various well-known problems were examined with respect to their expected appropriateness in terms of modelling the THSP within the various contexts mentioned above. This was done to exploit the available literature for possible angles not yet considered during the formulation of the scheduling model building block within the final DSS put forward in this dissertation.

Contribution 3 *A sequential model formulation in §6.2 for the scheduling model building block of the DSS.*

The formulation presented in §6.2 constitutes a novel (within the field of sugarcane harvest scheduling) and well-grounded scheduling model building block suitable for the THSP within the medium-scale solitary commercial grower and medium-scale commercial harvesting group contexts. A solution approach towards solving this scheduling model was described in §6.2.2 and it was shown that it is a very hard problem to solve in §6.2.1.

Contribution 4 *A proposed modelling framework in §7.4.2 for improving the forecast of seasonal cane yield for sugarcane crops that have been affected by one or several extraneous events.*

The framework of the regression-based *base yield models* in §7.4.1 coupled with the *event-driven yield models* in §7.4.2 was designed to improve the forecast of seasonal cane yield with respect to crops that have been exposed to extraneous events. The base yield models are not novel, but the event yield models and the coupling of these models are novel.

Contribution 5 *A proposed modelling framework in §7.4.2 for improving the forecast of seasonal recoverable value percentage for sugarcane crops that have been affected by one or several extraneous events.*

The framework of the regression-based *base RV models* in §7.4.3 coupled with the *event-driven RV models* in §7.4.4 was designed to improve the forecast of seasonal cane RV with respect to crops that have been exposed to extraneous events. The base RV models are not novel, but the event RV models and the coupling of these models are novel.

Contribution 6 *An appraisal of environmental events and human-induced events in §2.6 and §2.7 with a proposed quantification of their effects with respect to the yield and quality of sugarcane, presented in Table 7.3*, which fits into the framework described in §7.4.

The gathering and organisation of information available in the literature concerning environmental and other events that affect sugarcane adversely has not previously been presented in the literature. The information found was analysed in order to quantify the effects of these events on the yield and quality of sugarcane. The actual coefficient values assigned in Table 7.3 are proposals, not experimental results or known facts, based on §2.6, §2.7 and communication with field experts.

Contribution 7 *A cost modelling framework in §7.5 designed to further the accuracy of seasonal profit forecasts of sugarcane fields, especially with respect to extraneously affected fields.*

In §7.5.1, §7.5.2 and §7.5.3, a framework for properly costing sugarcane field harvesting as well as selectively penalising undesirable harvesting dates due to field properties, extraneous events or combinations thereof, was described. This is novel.

Contribution 8 *The DSS architecture in §8.1.*

The main contribution of this dissertation is the DSS architecture described in §8.1 designed through its various building blocks to predict the values of sugarcane fields into the future, schedule the fields in profitable sequences and provide these schedules in support of decision making within sugarcane harvesting operations. The DSS architecture as described in §8.1 is a novel manner in which to approach the sugarcane harvest scheduling problem.

Contribution 9 *The computer implementation of the DSS architecture in §8.2.*

The computer implementation of the DSS architecture described in §8.2.1 and §8.2.2 was verified in §8.3 and its parameter settings were fine-tuned in §8.4. The implementation itself is a novel contribution to the currently available computer programs that can provide decision support to managers of sugarcane harvesting operations.

Contribution 10 *The validation experiments in §9.1 and §9.2.*

Finally, this DSS architecture was validated to provide "acceptable" to "very good" schedules by industry experts, through a series of validation experiments performed by shadow scheduling an actual harvesting operation via an early implementation in §9.1 and the final DSS implementation in §9.2. These validation experiments are novel in terms of what has previously been conducted in South Africa with respect to sugarcane harvest scheduling decision support tool validation experiments, as far as the author can ascertain.

10.3 An appraisal of the dissertation contributions

The definition of the THSP in Chapter 4 is of value since it provides a problem formulation general enough to invite several methodologies for its solution, while it is well-defined enough to provide a point of reference for comparing these various methodologies. Today, there are many methodological comparisons in the literature on sugarcane production, but these sometimes compare different methodologies across different problems, leading to possible misconceptions about the value of the methodologies. Therefore, the definition in Chapter 4 may improve the comparisons of future research in seasonal sugarcane harvest scheduling. If the THSP is recognised by the scientific community as a problem worthy of their attention, the formulation in Chapter 4 may contribute significantly to the development of the field.

The exploration in §5.1 of well-known mathematical programming models and their relationships with the THSP within its various contexts provides a number of viewpoints to be taken when formulating optimisation models for sugarcane harvest scheduling problems. These viewpoints may constitute starting points for future research on the THSP.

The sequential model formulation in §6.2 and its solution in §6.2.2 provides a valid (Chapter 9) manner in which to sequence (or schedule) a set of sugarcane fields based on their forecast time-dependent seasonal profits.

The modelling of extraneous events in §7.4 and in §7.5 in terms of cane yield, RV and cost penalties was shown by means of a hindsight schedule in §9.2.4 to provide, in a single case, an improvement in SPD-2 from ninety to fifty-eight days. This single sample constitutes no statistical evidence, but it does provide an indication of the improvement potential arising as a result of the incorporation of the event-driven yield and RV models. The comments collected during the 2009 and the 2010 validation experiments further indicate that these considerations are paramount in scheduling the harvesting operation.

The most important contribution of this dissertation is the validated DSS implementation in §8.2, whose validation reflects on the decision support system architecture in §8.1 as being an acceptable solution to the THSP for medium-scale solitary commercial growers and medium-scale commercial harvesting groups formulated in Chapter 4. The validation experiments showed that the DSS implementation delivered—according to the opinions of industry experts—one poor schedule, twelve acceptable schedules, seven good schedules and four very good schedules during the 2009 validation experiment. Hindsight schedules in §9.2.1 prove statistically that the DSS-generated schedules are better than random schedules as predictors of the actual harvesting sequence. The final comments in §9.2.4 after the 2010 validation experiment includes the belief, expressed by several of the growers, that the schedules would be useful if their operations were much larger, in the range of 25 000 tonnes of cane per annum and upwards.

The sugar industry may benefit from this research, since the DSS may be the first automated harvest scheduling system that works for a large variety of sugarcane growers in South Africa. It may be used as a negotiating tool within harvesting groups who may seek an “impartial” system to decide the harvesting sequence or it may provide decision support to managers of large-scale harvesting operations.

CHAPTER 11

Recommendations for future research

The harvest scheduling problem described in this dissertation is in its nature contextual, and more research is required in order to understand the problem and in order to model it further. Several components of the building blocks of the DSS architecture are in an early stage of development and may increase in quality and prediction accuracy by means of further study. A number of avenues of future research intended to spawn future work on this intriguing problem, are proposed in this final chapter. The first four proposals concern the prediction models building block, the following four proposals relate to the formulation and solution of the scheduling model and the last two proposals deal with an improved DSS architecture. This chapter stands in fulfilment of Dissertation Objective IX of §1.3.

Proposal 1 *To develop empirical models of the effects of extraneous events on cane yield and RV over time.*

The potentially vast undertaking of modelling all extraneous events that may affect the yield or quality of sugarcane may be carried out by means of a series of experiments. The factors involved in these experiments may for a particular event, include:

- the time of year at the onset of the event,
- the soil properties of the field (*i.e.* several factors),
- the cane variety (varieties may perhaps be grouped according to already known behaviour in response to the event),
- the age of the cane in the field,
- the field aspect and the field toposequence,
- latitude, longitude and altitude of the field,
- whether dry or normal conditions prevailed before the onset of the event,
- whether dry or normal conditions prevailed during the experiment.

A nation-wide experiment (or series of experiments) with factors that may be controlled, as well as factors that may only be recorded, may shed more light on the relationships between these factors and their associated rates of deterioration of sugarcane.

Proposal 2 *Use simulation-based crop prediction models in a standalone fashion.*

Simulation-based crop prediction models are well-developed in the literature and it may be possible to combine these models within a standalone DSS implementation. During this dissertation project, SASRI showed that they are helpful, should one decide to attempt to use their model (CANEGRO) to predict various important base properties of sugarcane. The potential benefits are large in terms of increasing the base yield and base RV model accuracies.

Proposal 3 *Use heat units instead of time as a regressor in the cane yield models.*

Cane yield may be modelled more precisely using the factor heat units rather than the factor time spent growing as regressor in the base yield models. This is thus an avenue that may provide better prediction models as a result of a relatively small amount of work invested.

Proposal 4 *Improve the cost models in §7.5.*

More may be done in terms of modelling costs accurately, especially towards making it feasible to integrate the DSS architecture into a business information system. Much has been done in this area and most of the costing should be possible to mine from the literature.

Proposal 5 *Incorporate restricted adjacency harvesting in the case of twelve-month crops.*

The harvesting of fields should, if possible, be performed in such a way that neighbouring fields retain a significant age difference [25]. When fields are left in an age mosaic pattern, erosion due to rain on hill slopes is reduced. The risk of fires wreaking havoc on adjacent fields is also reduced, since fires are more easily extinguished in young cane than in old cane [164]. The harvesting of certain *restricted pairs* of fields within a certain number of time periods from one another may be forbidden by augmenting the SMF (problem (6.7)–(6.19)) with additional constraints. One possibility is to employ the ATSPTDC formulation in §5.1.7, which will accommodate adjacency control by means of time windows whereby one could assign a time window to each restricted field. There is, however, a drawback here, since one would have to choose beforehand which of the two fields in each pair should be assigned to which harvesting time window, thereby deciding the internal harvesting sequence within each restricted pair.

In terms of the formulation in §6.2.2, one may alter the objective function to incorporate a penalty for harvesting the restricted pairs within a certain number of time periods from one another. It is a matter of computing the differences in harvesting times for all restricted pairs and penalising those that are less than a certain value, determined beforehand. The penalty constants, of course, need to be fine-tuned, before the algorithm may be put to use.

Proposal 6 *Incorporate a distance matrix into the cost components of §7.5 in order to account for travelling costs and to restrict certain harvesting front movements.*

If the objective function of the SMF in §6.2.1 is changed to

$$\text{maximise} \qquad z = \sum_{u\in I}\sum_{v\in I}\sum_{j\in J}(P_{uj}x^{\delta}_{uvj} + C^{d}_{uvj}x^{\delta}_{uvj}), \tag{11.1}$$

where C^d_{uvj} is the cost of travelling from field u to field v starting at time instant j, then it is possible to account for travelling costs and also to enforce certain adjacency restrictions by means of artificial penalties within C^d_{uvj}. This change would also not be computationally demanding for the tabu search, in a relative sense, since the current objective function already has to compute the time instants of the harvesting commencement of every field. One interesting idea is to penalise travelling more during the wet periods of the season, since travelling is more difficult then than during the dry periods of the season. This notion has not been discussed with the industry experts, but should provide an interesting topic.

In some scenarios applicable to the decision support system, the geographical situation of the fields under consideration might be such that they can be grouped into a number of separate *general areas*. An example of such a situation is when a large farm consists of two or more groups of fields that are separated by a large distance. To move the harvesting front, which consists of all the cutters or mechanical harvester, loading machinery, water carts and any other equipment and personnel required to burn and harvest a particular field, between general areas may come at a high cost, and should be avoided if possible. In order to restrict such movement, one may utilise the C^d_{uvj}-parameters in (11.1) and penalise all arcs leading from one general area to the next to some extent.

Proposal 7 *Implement a better upper bounding technique on the objective function value in §6.2.2 into the scheduling model building block.*

Currently, the maximum profit possible for each field is taken, and these profits are summed into an upper bound on the objective function value. For certain instances of the problem, this upper bound is very poor. Using the linear programming relaxation of the formulation in §6.2.1 is one option, but removing certain constraints first and then linearly relaxing the resulting problem may be a better option, due to the large number of constraints present even in moderate size problem instances.

Proposal 8 *Explore other methods of solving problem* (6.7)–(6.19) *approximately.*

The body of solution methods for combinatorial optimisation problems in the literature can probably offer faster and stronger solution methods than the one (*i.e.* tabu search) implemented in this dissertation. It is therefore suggested that such work is undertaken, perhaps for the SMF augmented with a distance matrix as well.

Proposal 9 *Test and develop the DSS further, and integrate it with other business information systems.*

Many questions about the DSS may still be unanswered and some of these questions may be addressed by testing the DSS in several distinct regions, for several years and even in other countries. Opportunities for further improvement of the DSS will undoubtedly arise during such tests. In order to pursue this proposal, a larger project is necessary, involving computer and decision support systems scientists, business information systems scientists, GIS experts, logisticians, social scientists and others, depending on the set goals of such a project.

Proposal 10 *Redesign or adapt the DSS architecture to accommodate the large-scale independent estate and large-scale mill-owned estate contexts of the THSP described in §4.1.3.*

The DSS may be adapted to the large-scale independent estate and large-scale mill-owned estate contexts by shadow scheduling several estates across a season. Such an endeavour should be completed in close collaboration with the managers and owners of such estates. The results may lead to a different DSS architecture altogether, or may only result in additions or changes to the one presented in §8.1.

References

[1] Achutan NR & Hardjawidjaja A, 2001, *Project scheduling under time dependent costs—A branch and bound algorithm*, Annals of Operations Research, **108**, pp. 55–74.

[2] Almazan O, Gonzalez L & Galvez L, 1998, *The sugarcane, its by-products and co-products*, [Online], [Cited August 28th, 2009], Available from `http://www.gov.mu/portal/sites/ncb/moa/farc/amas98/keynote.pdf`

[3] Alonzo-Pippo W, Luengo CA, Koehlinger J, Garzone P & Cornacchia G, 2008, *Sugarcane energy use: The Cuban case*, Energy Policy, **36**, pp. 2163–2181.

[4] Applegate AL, Bixby RE, Chvátal V, Cook W, Espinoza DG, Goycoolea M & Helsgaun K, 2009, *Certification of and optimal tour through 85,900 cities*, Operations Research Letters, **37**, pp. 11–15.

[5] Albiach J, Sanchis JM & Soler D, 2008, *An asymmetric TSP with time windows and with time-dependent travel times and costs: An exact solution through a graph transformation*, European Journal of Operational Research, **189**, pp. 789–802.

[6] Aramyan LH, Oude Lansink AGJM, van der Vorst JGAJ & van Kooten O, 2007, *Performance measurement in agri-food supply chains: A case study*, Supply Chain Management, **12:4**, pp. 304–315.

[7] ARENA Simulation Software, 2010, *ARENA simulation software by Rockwell Automation*, [Online], [Cited July 6th, 2010], Available from `http://www.arenasimulation.com/`

[8] Balas E, Ceria S, Cornéjols G & Natraj N, 1996, *Gomory cuts revisited*, Operations Research Letters, **19:1**, pp. 1–9.

[9] Balinski ML, 1985, *Signature methods for the assignment problem*, Operations Research, **33:3**, pp. 527–536.

[10] Bazley D & Bazley MR, 2000, *Nil Desperandum—the Bazley story*, 1st Edition, The Royle Trust, Kloof, South Africa.

[11] Albiach J, Sanchis JM & Soler D, 1999, *Measuring supply chain performance*, International Journal of Operations & Production Management, **19:3**, pp. 275–292.

[12] Belfiore P, Tsugunobu H & Yoshizaki Y, 2009, *Scatter search for a real-life heterogeneous fleet vehicle routing problem with time windows and split deliveries in Brazil*, European Journal of Operational Research, **199:3**, pp. 750–758.

[13] Beraldi P, Ghiani G, Laporte G & Musmanno R, 2005, *Efficient neighbourhood search for the probabilistic pickup and delivery travelling salesman problem*, Networks, **45**, pp. 195–198.

[14] Berding N & Hurney AP, 2005, *Flowering and lodging, physiological-based traits affecting cane and sugar yield—What do we know of their control mechanisms and how do we manage them?*, Field Crops Research, **92**, pp. 261–275.

[15] Best of Sicily Magazine, 2010, *Sugar cane in Sicily*, [Online], [Cited June 22$^{\text{nd}}$, 2010], Available from `http://www.bestofsicily.com/mag/art143.htm`

[16] Bezuidenhout CN & Singels A, 2007, *Operational forecasting of South African sugarcane production: Part 1—System description*, Agricultural Systems, **92**, pp. 23–38.

[17] Bezuidenhout CN & Singels A, 2007, *Operational forecasting of South African sugarcane production: Part 2—System evaluation*, Agricultural Systems, **92**, pp. 39–51.

[18] Bezuidenhout CN, Singels A & Hellman D, 2002, *Whole farm harvesting strategy optimisation using the CANEGRO model: A case study for irrigated and rainfed sugarcane*, Proceedings of the South African Sugar Technologists' Association, **76**, pp. 250–259.

[19] Bichou K & Gray R, 2004, *A logistics and supply chain management approach to port performance measurement*, Maritime Policy and Management, **31:1**, pp. 47–67.

[20] Borowski EJ & Borwein JM (Eds), 2002, *Collins Dictionary of Mathematics*, Harper Collins Publishers, Glasgow.

[21] Bowman EH, 1956, *Production scheduling by the transportation method of linear programming*, Operations Research, **4**, pp. 100–103.

[22] Bowman EH, 1959, *The schedule-sequencing problem*, Operations Research, **7:5**, pp. 621–624.

[23] Box GEP & Draper NR, 1987, *Empirical model building and response surfaces*, John Wiley & Sons, New York (NY).

[24] Box GEP, Hunter WG & Hunter JS, 1979, *Statistics for experimenters*, John Wiley & Sons, New York (NY).

[25] Bruggeman E, 2008, Extension Officer at *Eston Canegrowers*, [Personal Communication], Contactable at `ebruggemann@illovo.co.za`.

[26] Cadet P, Berry SD, Leslie GW & Spaull VW, 2007, *Management of nematodes and a stalk borer by increasing within-field sugarcane cultivar diversity*, Plant Pathology, **56**, pp. 526–535.

[27] Carlson J, Jaffe A & Wiles A (Eds), 2006, *The millenium prize problems*, American Mathematical Society, Providence (RI).

[28] Carnegie AJM & Smaill RJ, 1982, *Pre-trashing of sugarcane as a means of combating the borer* Eldana saccharina *Walker*, Proceedings of the South African Sugar Technologists' Association, **56**, pp. 78–81.

[29] Carpaneto G, Dell'Amico M & Toth P, 1995, *Exact solution of large-scale, asymmetric traveling salesman problems*, ACM Transactions on Mathematical Software (TOMS), **21:4**, pp. 394–409.

[30] Chan FTS & Qi HJ, 2003, *An innovative performance measurement method for supply chain management*, Supply Chain Management, **8:3**, pp. 209–223.

[31] Chan FTS & Qi HJ, 2003, *Feasibility of performance measurement system for supply chain: A process-based approach and measures*, Integrated Manufacturing Systems, **14:3**, pp. 179–190.

[32] Cheng CB & Mao CP, 2007, *A modified ant colony system for solving the travelling salesman problem with time windows*, Mathematical and Computer Modelling, **46**, pp. 1225–1235.

[33] Cheng R & Gen M, 1997, *Parallel machine scheduling problems using memetic algorithms*, Computers & Industrial Engineering, **33:3–4**, pp. 761–764.

[34] Chewing Cane, 2010, *History of sugar cane*, [Online], [Cited June 22$^{\text{nd}}$, 2010], Available from `http://www.chewingcane.com/sugarcane_history.html`

[35] Chiang TC, Cheng HC & Fu LC, 2010, *A memetic algorithm for minimizing total weighted tardiness on parallel batch machines with incompatible job families and dynamic job arrival*, Computers & Operations Research, **37**, pp. 2257–2269.

[36] Chohfi FM, 2008, *Discussion of the routes and potential for dimethyl ether production from the biomass available in the sugarcane value chain*, Energy for Sustainable Development, **12:1**, pp. 62–64.

[37] Choi IC, Kim SI & Kim HS, 2003, *A genetic algorithm with a mixed region search for the asymmetric traveling salesman problem*, Computers & Operations Research, **30**, pp. 773–786.

[38] Clowes M St. J & Wood RA, 1978, *Post-harvest deterioration of whole stalk sugarcane treated with chemical ripeners*, Proceedings of the South African Sugar Technologists' Association, **60**, pp. 166–168.

[39] Cook WJ, Cunningham WH, Pulleyblank WR & Schrijver A, 1998, *Combinatorial optimization*, John Wiley & Sons, New York (NY).

[40] Coulthard C, 2008, Sugarcane grower in the *Eston Mill area*, [Personal Communication], Contactable at `cottonwood@absamail.co.za`.

[41] Crowe KA & Nelson JD, 2005, *An evaluation of the simulated annealing algorithm for solving the area-restricted harvest scheduling model against optimal benchmarks*, Canadian Journal of Forestry Research, **35**, pp. 2500–2509.

[42] Daniel C, 1959, *Use of half-normal plots in interpreting factorial two level experiments*, Technometrics, **1**, pp. 311–342.

[43] Danna E, Rothberg E & Le Pape C, 2005, *Exploring relaxation induced neighbourhoods to improve MIP solitions*, Mathmatical Programming, Series A, **102**, pp. 71–90.

[44] Dantzig GB, 1982, *Reminiscences about the origins of linear programming*, Operations Research Letters, **1:2**, pp. 43–48.

[45] Dantzig GB, Fulkerson R & Johnson S, 1954, *Solution of a large-scale traveling-salesman problem*, Journal of the Operations Research Society of America, **2**, pp. 393–410.

[46] Dantzig GB & Ramser JH, 1959, *The truck dispatching problem*, Management Science, **6**, pp. 80–91.

[47] De Lange J, 2008, Cane Procurement Manager at *Noodsberg Mill*, [Personal Communication], Contactable at jdlange@illovo.co.za.

[48] Department of Trade and Industry, Republic of South Africa, 2009, *Publications*, [Online], [Cited February 18th, 2009], Available from http://www.thedti.gov.za/publications/SugarAct/4DeregulationoftheSugar.pdf

[49] Department of Trade and Industry, Republic of South Africa, 2000, *Sugar Industry Agreement*, Unpublished document.

[50] DePuy GW, Moraga RJ & Whitehouse GE, 2003, *Meta-RaPS: a simple and effective approach for solving the traveling salesman problem*, Transportation Research Part E, **41**, pp. 115–130.

[51] Díaz JA & Fernández E, 2001, *A tabu search heuristic for the generalized assignment problem*, European Journal of Operational Research, **132**, pp. 22–38.

[52] Dorigo M, Maniezzo V & Colorni A, 1996, *The ant system: optimization by a colony of cooperating agents*, IEEE Transactions on Systems, Man and Cybernetics—Part B, **26:1**, pp. 1–13.

[53] Eggleston G & Legendre B, 2003, *Mannitol and oligosaccharides as new criteria for determining cold tolerance in sugarcane varieties*, Food Chemistry, **80**, pp. 451–461.

[54] Ethnobotanical Leaflets, 2010, *Ethnobotanical leaflets*, [Online], [Cited June 22nd, 2010], Available from http://www.ethnoleaflets.com/leaflets/sugar.htm

[55] Financial Times, 2010, *FT.com / commodities – sugar surplus knocks prices*, [Online], [Cited June 21st, 2010], Available from http://www.ft.com/cms/s/0/f2ed2a66-5ebd-11df-af86-00144feab49a.html

[56] Fisher M, 1981, *The Lagrangian relaxation method for solving integer programming problems*, Management Science, **27**, pp. 1–18.

[57] Fisher ML, Jaikumar R & van Wassenhove LN, 1986, *A multiplier adjustment method for the generalized assignment problems*, Management Science, **32:9**, pp. 1095–1103.

[58] Fleming GF, 2006, *Cooperative systems: An information systems model for industry value chain management*, International Sugar Journal, **108**, p. 299.

[59] Fox KR, Gavish B & Graves SC, 1980, *An n-constraint formulation of the (time dependent) travelling salesman problem*, Operations Research, **28**, pp. 1018–1021.

[60] França KR, Mendes A & Pablo M, 2001, *A memetic algorithm for the total tardiness single machine scheduling problem*, European Journal of Operational Research, **132**, pp. 224–242.

[61] Gagné C, Price WL & Gravel M, 2002, *Comparing an ACO [ant colony optimisation] algorithm with other heuristics for the single machine scheduling problem with sequence-dependent setup times.*, Journal of the Operational Research Society, **53**, pp. 1018–1021.

[62] Galloway JH, 1989, *The sugar cane industry: an historical geography from its origins to 1914*, 2nd Edition, Cambridge University Press, New York.

[63] GAMS Development Corporation, 2010, *GAMS home page*, [Online], [Cited June 25th, 2010], Available from `http://www.gams.com/`

[64] García-Martínez C, Cordón O & Herrera F, 2007, *A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP*, European Journal of Operational Research, **180**, pp. 116–148.

[65] Gascho GJ & Miller JD, 1979, *Post-freeze deterioration of six sugarcane cultivars*, Agronomy Journal, **71**, pp. 275–278.

[66] Gaucher S, Le Gal PY & Soler G, 2008, *Modelling supply chain management in the sugar industry*, Proceedings of the South African Sugar Technologists' Association, **7**, pp. 542–554.

[67] Gauss CF, 1906, *Theoria motus corporum coelestium in sectionibus conicis solem ambientium / auctore Carolo Friderico Gauss*, Königlische Gesellschaft der Wissenschaften, Leipzig.

[68] Giles RC, Bezuidenhout CN & Lyne PWL, 2008, *Evaluating the feasibility of a sugarcane vehicle delivery schedule: A theoretical study*, International Sugar Journal, **110**, pp. 242–247.

[69] Giles RC, Dines GR, Lyne PWL & Bezuidenhout CN, 2006, *The complexities of introducing the FREDD vehicle scheduling system into the Darnall mill area*, Proceedings of the South African Sugar Technologists' Association, **80**, pp. 66–70.

[70] Gilmour P, 1999, *A strategic audit framework to improve supply chain performance*, Journal of Business and Industrial Marketing, **14:5**, pp. 355–363.

[71] Gomory R, 1958, *Outline of an algorithm for integer solutions to linear programs*, Bulletin of the American Mathematical Society, **64**, pp. 275–278.

[72] Glover F, 1977, *Heuristics for integer programming using surrogate constraints*, Decision Sciences, **8:1**, pp. 156–166.

[73] Glover F, 1989, *Tabu search—part I*, ORSA Journal on Computing, **1:3**, pp. 190–206.

[74] Glover F, 1990, *Tabu search—part II*, ORSA Journal on Computing, **2:1**, pp. 4–32.

[75] Glover F, 1996, *Ejection chains, reference structures and alternating path methods for traveling salesman problems*, Discrete Applied Mathematics, **65**, pp. 223–253.

[76] Goebel FR, Way MJ & Gossard C, 2005, *The status of* Eldana saccharina *(Lepidoptera: pyralidae) in the South African sugar industry based on regular survey data*, Proceedings of the South African Sugar Technologists' Association, **79**, pp. 337–346.

[77] Goebel FR & Way MJ, 2003, *Investigation of the impact of* Eldana saccharina *(Lepidoptera: pyralidae) on sugarcane yield in field trials in Zululand*, Proceedings of the South African Sugar Technologists' Association, **77**, pp. 256–265.

[78] Goldemberg J & Guardabassi P, 2009, *Are biofuels a feasible option?*, Energy Policy, **37**, pp. 10–14.

[79] Greenland D, 2005, *Climate variability and sugarcane yield in Louisiana*, Journal of Applied Meteorology, **44**, pp. 1655–1666.

[80] Gross D & Harris CM, 1998, *Fundamentals of queueing theory*, 3rd Edition, Wiley Interscience, New York (NY).

[81] Grunow M, Günther HO & Westinner R, 2007, *Supply optimization for the production of raw sugar*, International Journal of Production Economics, **110**, pp. 224–239.

[82] Grötschel M & Holland O, 1991, *Solution of large-scale travelling salesman problems*, Mathematical Programming, **51**, pp. 141–202.

[83] Guilleman E, Le Gal PY, Meyer E & Schmidt E, 2003, *Assessing the potential for improving mill area profitability by modifying cane supply and harvest scheduling—A South African study*, Proceedings of the South African Sugar Technologists' Association, **77**, pp. 566–579.

[84] Gunasekaran A, Patel C & McGaughey RE, 2004, *A framework for supply chain performance measurement*, International Journal of Production Economics, **87**, pp. 333–347.

[85] Gunasekaran A & Tirtiroglu E, 2001, *Performance measures and metrics in a supply chain environment*, International Journal of Operations & Production Management, **21:1**, pp. 71–87.

[86] Haldar S, 1992, *Subset selection in regression*, Journal of Marketing Research, **29:2**, pp. 270–272.

[87] Hansen AC, Barnes AJ & Lyne PWL, 2002, *Simulation modeling of sugarcane harvest-to-mill delivery systems*, Transactions of the American Society for Agricultural Engineers, **45**, pp. 531–538.

[88] Harries L, 2008, Chairman of *Noodsberg Canegrowers*, [Personal Communication], Contactable at +27 82 789 25 45.

[89] Hassan SH & Nasr MI, 2008, *Sugar industry in Egypt*, Sugar Tech, **10:3**, pp. 204–209.

[90] Held M & Karp RM, 1971, *The travelling-salesman problem and minimum spanning trees: Part II*, Mathematical Programming, **1**, pp. 6–25.

[91] Helsgaun K, 2000, *An effective implementation of the Lin-Kernighan TSP heuristic*, European Journal of Operational Research, **126**, pp. 106–130.

[92] Helsgaun K, 2009, *General k-opt submoves for the Lin-Kernighan TSP heuristic*, Mathematical Programming Computation, **1:2–3**, pp. 119–163.

[93] Hervani AA, Helms MM & Sarkis J, 2005, *Performance measurement for green supply chain management*, Benchmarking: An International Journal, **12:4**, pp. 330–353.

[94] Higgins AJ, 1999, *Optimizing cane supply decisions within a sugar mill region*, Journal of Scheduling, **2**, pp. 229–244.

[95] Higgins AJ, 2001, *A dynamic tabu search for large-scale generalised assignment problems*, Computers & Operations Research, **28**, pp. 1039–1048.

[96] Higgins AJ, 2002, *Australian sugar mills optimise harvester rosters to improve production*, Interfaces, **32:3**, pp. 15–25.

[97] Higgins AJ, 2006, *Scheduling of road vehicles in sugarcane transport: A case study at an Australian sugar mill*, European Journal of Operational Research, **170**, pp. 987–1000.

[98] Higgins AJ, Antony G, Sandell G, Davies I, Prestwidge D & Andrew B, 2004, *A framework for integrating a complex harvesting and transport system for sugar production*, Agricultural Systems, **82**, pp. 99–115.

[99] Higgins AJ, Archer A, Jakku E, Thorburn E & Prestwidge E, 2005, *Value chain research in sugar: A review of successes and learnings*, (Unpublished) Technical Report, CSIRO Sustainable Ecosystems, St. Lucia (QLD 4067), Australia.

[100] Higgins AJ & Davies I, 2005, *A simulation model for capacity planning in sugarcane transport*, Computers and Electronics in Agriculture, **47**, pp. 85–102.

[101] Higgins AJ, Muchow RC, Rudd AV & Ford AW, 1998, *Optimising harvest date in sugar production: A case study for the Mossman mill region in Australia I. Development of operations research model and solution*, Field Crops Research, **57**, pp. 153–162.

[102] Higgins AJ & Postma S, 2004, *Australian sugar mills optimise siding rosters to increase profitability*, Annals of Operations Research, **128**, pp. 235–249.

[103] Higgins AJ, Thorburn P, Archer A & Jakku E, 2007, *Opportunities for value chain research in sugar industries*, Agricultural Systems, **94**, pp. 611–621.

[104] Holland JH, 1962, *Outline for a logical theory of adaptive systems*, Journal of the Association for Computing Machinery, **9:3**, pp. 297–314.

[105] Hvattum LM & Glover F, 2009, *Finding local optima of high-dimensional functions using direct search methods*, European Journal of Operational Research, **195:1**, pp. 31–45.

[106] Iannoni AP & Morabito R, 2006, *A discrete simulation analysis of a logistics supply system*, Transportation Research Part E, **42**, pp. 191–210.

[107] IBM, 2010, *IBM ILOG CPLEX optimizer*, [Online], [Cited June 25th, 2010], Available from `http://www-304.ibm.com/jct01003c/software/integration/optimization/cplex-optimizer/`

[108] Illovo, 2010, *Illovo Sugar – South Africa*, [Online], [Cited June 23rd, 2010], Available from `http://www.illovosugar.com/World_of_sugar/Sugar_Statistics/South_Africa.aspx`

[109] Inman-Bamber NG, 1991, *A growth model for sugar-cane based on a simple carbon balance and the CERES-Maize water balance*, South African Journal of Plant Soil, **8**, pp. 93–99.

[110] Inman-Bamber NG, Culverwell TL & McGlinchey MG, 1993, *Predicting yield responses to irrigation of sugarcane from a growth model and field records*, Proceedings of the South African Sugar Technologists' Association, **67**, pp. 66–72.

[111] Inman-Bamber NG, Muchow RC & Robertson MJ, 2002, *Dry matter partitioning of sugarcane in Australia and South Africa*, Field Crops Research, **76:1**, pp. 71–84.

[112] Inman-Bamber NG & Smith DM, 2005, *Water relations in sugarcane and response to water deficits*, Field Crops Research, **92**, pp. 185–202.

[113] Irvine JE, 1967, *Effects of an early freeze on Louisiana sugarcane*, Proceedings of the American Society of Sugarcane Technologists, **14**, pp. 10–15.

[114] i Six Sigma, 2009, *Six Sigma—What is Six Sigma?*, [Online], [Cited September 28th, 2009], Available from `http://www.isixsigma.com/sixsigma/six_sigma.asp`

[115] Jayalakshmi GA & Sathiamoorthy S, 2001, *A hybrid genetic algorithm—A new approach to solve traveling salesman problem*, International Journal of Computational Engineering Science, **2**, pp. 339–355.

[116] Jiao Z, Higgins AJ & Prestwidge DB, 2005, *An integrated statistical and optimisation approach to increasing sugar production within a mill region*, Computers and Electronics in Agriculture, **48**, pp. 170–181.

[117] Karlsson J, Rönnqvist M & Bergström J, 2003, *Short-term harvest planning including scheduling of harvest crews*, International Transactions in Operational Research, **10**, pp. 413–131.

[118] Keating BA, Robertson MJ, Muchow RC, Huth NI, 1999, *Modelling sugarcane production systems. 1. Development and performance of the sugarcane module*, Field Crops Research, **61**, pp. 253–271.

[119] Keeping MG & Rutherford RS, 2004, *Resistance mechanisms of South African sugarcane to the stalk borer* Eldana saccharina *(Lepidoptera: Pyralidae): A review*, Proceedings of the South African Sugar Technologists' Association, **78**, pp. 307–312.

[120] Kirkpatrick S, Gelatt, Jr. CD & Vecchi MP, 1983, *Optimization by simulated annealing*, Science, **220**, pp. 671–680.

[121] Kleijnen JPC & Smits MT, 2003, *Performance metrics in supply chain management*, Journal of the Operational Research Society, **54**, pp. 1–8.

[122] Kohl N & Madsen OBG, 1997, *An optimization algorithm for the vehicle routing problem with time windows based on Lagrangian relaxation*, Operations Research, **45:3**, pp. 395–406.

[123] Kolahan F & Liang M, 1996, *A tabu search approach to optimization of drilling operations*, Computers & Industrial Engineering, **31**, pp. 371–374.

[124] Kovács A, Brown KN & Tarim A, 2009, *An efficient MIP model for the capacitated lot-sizing and scheduling problem with sequence-dependent setups*, International Journal of Production Economics, **118**, pp. 282–291.

[125] Kuhn HW, 1955, *The Hungarian method for the assignment problem*, Naval Research Logistics Quarterly, **2:1–2**, pp. 83–97.

[126] Kyle S, 2008, Operations Manager at *Mkhuzane Harvesting*, [Personal Communication], Contactable at `moniquemullin@hotmail.com`.

[127] Laguna M, Barnes JW & Glover FW, 1990, *Tabu search methods for a single machine scheduling problem*, Journal of Intelligent Manufacturing, **2**, pp. 63–74.

[128] Lai K, Ngai EWT & Cheng TCE, 2002, *Measures for evaluating supply chain performance in transport logistics*, Transportation Research Part E, **38**, pp. 439–456.

[129] Lambert AJD, 2006, *Exact methods in optimum disassembly sequence search for problems subject to sequence dependent costs*, Omega, **34**, pp. 538–549.

[130] Laporte G, 2007, *What you should know about the vehicle routing problem*, Naval Research Logistics, **54:8**, pp. 811–819.

[131] Larranaga P, Kuijpers CMH, Murga RH, Inza I & Dizdarevic S, 1999, *Genetic algorithms for the travelling salesman problem: A review of representations and operators*, Artificial Intelligence Review **13**, pp. 129–170.

[132] Lawes RA & Lawn RJ, 2005, *Application of industry information in sugarcane production systems*, Field Crops Research, **92**, pp. 353–363.

[133] Le Gal PY, Bezuidenhout CN & Lyne PWL, 2007, *Mill-scale supply chain and logistics model integration for improved decision support*, Proceedings of the International Society of Sugar Cane Technologists, **26**, pp. 121–130.

[134] Le Gal PY, Lejars C & Auzoux S, 2003, *MAGI: A simulation tool to address cane supply chain management issues*, Proceedings of the South African Sugar Technologists' Association, **77**, pp. 555–565.

[135] Le Gal PY, Lyne PWL, Meyer E & Soler LG, 2008, *Impact of sugarcane supply scheduling on mill sugar production: A South African case study*, Agricultural Systems, **96**, pp. 64–74.

[136] Legendre B, Tew T, Birkett H, Eggleston G, Finger C & Stein J, 2007, *Impact of subfreezing temperatures on the 2006 Louisiana sugarcane harvest*, Sugar Journal, **70:1**, pp. 27–28.

[137] Leslie GW, 2008, *Estimating the economic injury level and the economic threshold for the use of FASTAC against Eldana saccharina*, Proceedings of the South African Sugar Technologists' Association, **81**, pp. 319–323.

[138] Leslie GW, Stranack RA and de Haas O, 2006, *Progress in the use of aerially applied FASTAC (Alpha-cypermethrin) for the control of the sugarcane borer Eldana saccharina (Lepidoptera: pyralidae), and an assessment of its commercial impact*, Proceedings of the South African Sugar Technologists' Association, **80**, pp. 236–244.

[139] Lewis E, 2008, Sugarcane grower in the *Eston Mill area*, [Personal Communication], Contactable at `cindy@satweb.co.za`.

[140] LINGO, 2010, *Lindo Systems - optimization software: Integer programming, linear programming, nonlinear programming, global optimization*, [Online], [Cited June 10th, 2010], Available from `http://www.lindo.com/`

[141] Lin S & Kernighan BW, 1973, *An effective heuristic algorithm for the traveling-salesman problem*, Operations Research, **21**, pp. 498-516.

[142] Lionnet GRE, 1986, *Post-harvest deterioration of whole stalk sugarcane*, Proceedings of the South African Sugar Technologists' Association, **60**, pp. 52–57.

[143] Liu DL & Bull TA, 2001, *Simulation of biomass and sugar accumulation in sugarcane using a process-based model*, Ecological Modelling, **144**, pp. 181–211.

[144] Liu DL & Helyar KR, 2003, *Simulation of seasonal stalk water content and fresh weight yield of sugarcane*, Field Crops Research, **82**, pp. 59–73.

[145] Mak V & Boland N, 2007, *Polyhedral results and exact algorithms for the asymmetric travelling salesman problem with replenishment arcs*, Discrete Applied Mathematics, **155**, pp. 2093–2110.

[146] Martello S & Toth P, 1981, *An algorithm for the generalized assignment problem*, pp. 589–603 in Brans JP (Eds), *Operational Research 81*, North-Holland, Amsterdam.

[147] Martello S & Toth P, 1990, *Knapsack problems: Algorithms and computer implementations*, John Wiley & Sons, Chichester.

[148] Martin A & Sherington J, 1997, *Participatory research: Methods implementation, effectiveness and institutional context*, Agricultural Systems, **55**, pp. 195–216.

[149] Mathematica, 2010, *Wolfram Research: Mathematica, technical and scientific software*, [Online], [Cited May 9th, 2010], Available from `http://www.wolfram.com/`

[150] Matlab, 2010, *MathWorks - MATLAB and Simulink for technical computing*, [Online], [Cited May 9th, 2010], Available from `http://www.mathworks.com/`

[151] Mauritius Island Online, 2010, *Brief history*, [Online], [Cited June 22nd, 2010], Available from `http://www.maurinet.com/about_mauritius/brief_history`

[152] Miller AJ, 1990, *Subset selection in regression*, Chapman and Hall, New York.

[153] Miller CE, Tucker AW & Zemlin RA, 1960, *Integer programming formulation of traveling salesman problems*, Journal of the Association for Computing Machinery, **7:4**, pp. 326–329.

[154] Miller DL & Pekny JF, 1991, *Exact solution of large asymmetric traveling salesman problems*, Science, **251**, pp. 754–761.

[155] Minitab, 2010, *Statistical and process management software for Six Sigma and quality improvement*, [Online], [Cited May 9th, 2010], Available from `http://www.minitab.com/`

[156] Montgomery DC, 2005, *Design and analysis of experiments*, 5th edition, John Wiley & Sons, New York (NY)

[157] Montgomery DC, Peck EA & Vining GG, 2001, *Introduction to linear regression analysis*, John Wiley & Sons, New York (NY).

[158] Muchow RC, Higgins AJ, Rudd AV & Ford AW, 1998, *Optimising harvest date in sugar production: A case study for the Mossman mill region in Australia II. Sensitivity to crop age and crop class distribution*, Field Crops Research, **57**, pp. 243–251.

[159] Murray TJ, 2000, *The derivation of the RV formula for cane payment*, Paper presented at the Farming for RV Workshop of the South African Sugar Technologists' Association, Mt Edgecombe.

[160] Nagata Y, Bräysy O & Dullaert W, 2010, *A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows*, Computers & Operations Research, **37**, pp. 724–737.

[161] Nauss RM, 2003, *Solving the generalized assignment problem: An optimizing and heuristic approach*, INFORMS Journal on Computing, **15:3**, pp. 249–266.

[162] Nuss KJ, 1989, *Effects of flowering on sucrose content and sucrose yield in five sugarcane varieties*, Proceedings of the South African Sugar Technologists' Association, **63**, pp. 181–185.

[163] O'Leary GJ, 2000, *A review of three sugarcane simulation models with respect to their prediction of sucrose yield*, Field Crops Research, **68**, pp. 97–111.

[164] O'Neill R, 2008, Owner of *Seafield Farm*, [Personal Communication], Deceased.

[165] Park SE, Robertson M & Inman-Bamber NG, 2005, *Decline in the growth of a sugarcane crop with age under high input conditions*, Field Crops Research, **92**, pp. 305–320.

[166] Padberg M & Rinaldi G, 1991, *A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems*, SIAM Review, **33**, pp. 60–100.

[167] Pao DCW, Lam SP & Fong AS, 1999, *Parallel implementation of simulated annealing using transaction processing*, IEE Proceedings—Computers and Digital Techniques, **146**, pp. 107–113.

[168] Pentico DW, 2007, *Assignment problems: A golden anniversary survey*, European Journal of Operational Research, **176**, pp. 774–793.

[169] Petersen HL & Madsen OBG, 2009, *The double travelling salesman problem with multiple stacks—Formulation and heuristic solution approaches*, European Journal of Operational Research, **198**, pp. 139–147.

[170] Piewthongngam K, Pathumnakul S & Setthanan K, 2009, *Application of crop growth simulation and mathematical modeling to supply chain management in the Thai sugar industry*, Agricultural Systems, **102**, pp. 58–66.

[171] Pinedo ML, 2005, *Planning and scheduling in manufacturing and services*, Springer Science+Business Media, New York (NY).

[172] Plant Cultures, 2010, *Plant Cultures—sugar cane history*, [Online], [Cited June 18th, 2010], Available from `http://www.plantcultures.org/plants/sugar_cane_history.html`

[173] Poupaert E & Deville Y, 2000, *Simulated annealing with estimated temperature*, Artificial Intelligence Communications, **13**, pp. 19–26.

[174] Racer M & Amini MM, 1994, *A robust heuristic for the generalized assignment problem*, Annals of Operations Research, **50**, pp. 487–503.

[175] Rao PNG & Kumar KN, 2003, *Effect of flowering on juice quality and fibre content in sugarcane*, Sugar Tech, **5:3**, pp. 185–187.

[176] Ross GT & Soland RM, 1975, *A branch and bound algorithm for the generalized assignment problem*, Mathematical Programming, **8**, pp. 91–103.

[177] Rönnqvist M, 2003, *Optimization in forestry*, Mathematical Programming, **97**, pp. 267–284.

[178] Sahni S & Gonzales T, 1976, *P-complete approximation problems*, Journal of the Association for Computing Machinery, **23**, pp. 555–565.

[179] Saraiva AM, Hirakawa AR, Cugnasca CE, Pierossi MA & Hassuani SJ, 2000, *A weighing system for grab loaders for sugar cane yield mapping*, Precision Agriculture, **2**, pp. 293–309.

[180] SAS Institute, 2010, *SAS—Business analytics and business intelligence software*, [Online], [Cited May 9th, 2010], Available from `http://www.sas.com/`

[181] SASRI, 2010, *Farming Calendar*, [Online], [Cited June 20th, 2010], Available from `http://www.sasa.org.za/Farming_Calendar78.aspx`

[182] SASRI Info Pack, September 2008, [CD-ROM], South African Sugarcane Research Institute, Mount Edgecombe.

[183] SASRI, 2010, *SASRI overview*, [Online], [Cited May 14th, 2010], Available from `http://www.sasa.org.za/sasri_overview615.aspx`

[184] Seeruttun S, Barbe C & McIntyre G, 1999, *Crop cycle lengths and sugar yields*, Experimental Agriculture, **35:4**, pp. 407–416.

[185] Shillington (Ed), 2005, *The Encyclopedia of African History*, Taylor & Francis Group, New York (NY).

[186] Sinclair TR, Gilbert RA, Perdomo RE, Shine JM, Powell G & Montes G, 2005, *Volume of individual internodes of sugarcane stalks*, Field Crops Research, **91**, pp. 207–215.

[187] Singels A, Smit MA, Redshaw KA & Donaldson RA, 2005, *The effect of crop start date, crop class and cultivar on sugarcane canopy development and radiation interception*, Field Crops Research, **92**, pp. 249–260.

[188] Singh G, Chapman SC, Jackson PA & Lawn RJ, 2002, *Lodging reduces sucrose accumulation of sugarcane in the wet and dry tropics*, Australian Journal of Agricultural Research, **53**, pp. 1183–1195.

[189] SKIL, 2010, *SKIL – History of sugar*, [Online], [Cited June 22nd, 2010], Available from `http://www.sucrose.com/lhist.html`

[190] Soler D, Albiach J & Martínez E, 2009, *A way to optimally solve a time-dependent vehicle routing problem with time windows*, Operations Research Letters, **37**, pp. 37–42.

[191] Solomon S, 2004, *Post-harvest deterioration of sugarcane*, Sugar Tech, **11:2**, pp. 109–123.

[192] Solomon S & Li Y, 2004, *Chemical ripening of sugarcane: Global progress and recent developments in China*, Sugar Tech, **6**, pp. 241–249.

[193] SouthAfrica.info, 2010, *Hulett to pilot power project*, [Online], [Cited July 1st, 2010], Available from http://www.southafrica.info/business/economy/infrastructure/hulett-260208.htm

[194] South African Canegrowers, 2008, *Canegrowers*, [Online], [Cited February 18th, 2009], Available from http://www.sacanegrowers.co.za/

[195] South African Canegrowers, 2008, *RV price*, [Online], [Cited November 4th, 2008], Available from http://www.sacanegrowers.co.za/rvprice.htm

[196] South African History Online, 2010, *Men of dynamite*, [Online], [Cited June 22nd, 2010], Available from http://www.sahistory.org.za/pages/library-resources/online%20books/men-dynamite/chapter2.pdf

[197] South African History Online, 2010, *Stanger project*, [Online], [Cited June 22nd, 2010], Available from http://www.sahistory.org.za/pages/places/villages/kwazuluNatal/Stanger/chronology.htm

[198] SPSS, 2010, *SPSS, data mining, statistical analysis software, predictive analysis, predictive analytics, decision support systems*, [Online], [Cited May 9th, 2010], Available from http://www.spss.com/

[199] SQR Software, 2010, *CANEPRO – Cane management software*, [Online], [Cited June 22nd, 2010], Available from http://www.sqrsoftware.com/

[200] Sugarcane Technology, 2009, *RSD*, [Online], [Cited July 8th, 2010], Available from http://www.sugarcaneindia.com/RSD.htm

[201] Sugar India, 2010, *Sugar cane history*, [Online], [Cited June 22nd, 2010], Available from http://www.sugarindia.com/overview.asp

[202] Sugar Engineers, 2010, *Maps of sugar factories*, [Online], [Cited July 7th, 2010], Available from http://www.sugartech.co.za/factories/groupmap.php?cid=238

[203] Sugar Research and Development Corporation, 2005, *The value chain of the Australian sugar industry.* Technical report 1/2006, ISSN 1327-9475, [Australian Government Report], Sugar Research and Development Corporation, Brisbane.

[204] Stat-Ease, 2010, *Stat-Ease statistical software, training, and consulting for design of experiments (DOE)*, [Online], [Cited May 9th, 2010], Available from http://www.statease.com/

[205] Statistica, 2010, *Data mining, statistical analysis, software and services—StatSoft*, [Online], [Cited May 9th, 2010], Available from http://www.statsoft.com/

[206] Stecco G, Cordeau JF & Moretti E, 2009, *A branch-and-cut algorithm for a production scheduling problem with sequence-dependent and time-dependent setup times*, Computers & Operations Research, **12**, pp. 3–16.

[207] Stecco G, Cordeau JF & Moretti E, 2009, *A tabu search heuristic for a sequence-dependent and time-dependent scheduling problem on a single machine*, Journal of Scheduling, **35**, pp. 2635–2655.

[208] Stock JR, 2009, *A research review of supply chain management: Developments and topics for exploration*, Orion, **25:2**, pp. 147–160.

[209] Subhani MN, Chaudhry MA, Khaliq A & Muhammad F, 2008, *Efficacy of various fungicides againts sugarcane red rot* (Colletotrichum falcatum), International Journal of Agriculture & Biology, **10**, pp. 725–727.

[210] Suite101, 2010, *The history of sugar cane — Museum: Sugar cane, its history, Europe's last refining plant and museum*, [Online], [Cited June 22nd, 2010], Available from `http://spain-travel.suite101.com/article.cfm/sweet_tooth_for_sugar`

[211] Swinehart KD & Smith AE, 2005, *Internal supply chain performance measurement: A healthcare continuous improvement implementation*, International Journal of Health Care Quality Assurance, **18:7**, pp. 533–542.

[212] Tasgetiren MF, Pan QK & Liang YC, 2009, *A discrete differential evolution algorithm for the single machine total weighted tardiness problem with sequence dependent setup times*, Computers & Operations Research **36**, pp. 1900–1915.

[213] Thorburn PJ, Archer AA, Hobson PA, Higgins AJ, Sandel GR, Prestwidge DB & Antony G, 2008, *Evaluating regional net benefits of whole crop harvesting to maximise cogeneration*, International Sugar Journal **110**, pp. 632–637.

[214] Thompson M, 2008, Sugarcane grower in the *Eston Mill area*, [Personal Communication], Contactable at `mkt@iafrica.com`.

[215] Tongaat Hulett Sugar, 2010, *Tongaat Hulett — Operations — South Africa — Agriculture*, [Online], [Cited June 19th, 2010], Available from `http://www.huletts.co.za/ops/south_africa/agriculture.asp`

[216] Tripathi M, Agrawal S, Pandey MK, Shankar R & Tiwari MK, 2009, *Real world disassembly modeling and sequencing problem: Optimization by algorithm of self-guided ants (ASGA)*, Robotics and Computer-Integrated Manufacturing **25**, pp. 483–496.

[217] United States Department of Agriculture, 2008, *World centrifugal sugar production, supply and distribution*, [Online], [Cited February 17th, 2009], Available from `http://www.fas.usda.gov/htp/sugar/2008/May2008tables_sugar.pdf`

[218] University of Calgary, 1997, *European voyages of exploration: The sugar & slave trades*, [Online], [Cited June 22nd, 2010], Available from `http://www.ucalgary.ca/applied_history/tutor/eurvoya/Trade.html`

[219] Valente JMS & Goncalves JF, 2009, *A genetic algorithm approach for the single machine scheduling problem with linear earliness and quadratic tardiness penalties*, Computers & Operations Research, **36**, pp. 2720–2715.

[220] van Hoek RI, 1998, *"Measuring the unmeasurable"—Measuring and improving performance in the supply chain*, Supply Chain Management, **3:4**, pp. 187–192.

[221] van Hoek RI, 2001, *The contribution of performance measurement to the expansion of third party logistics alliances in the supply chain*, International Journal of Operations & Production Management, **21:1**, pp. 15–29.

[222] van Hoek RI, Harrison A & Christopher M, 2001, *Measuring agile capabilities in the supply chain*, International Journal of Operations & Production Management, **21:1**, pp. 126–147.

[223] Viswanathan R, Ramesh Sundar A, Malathi P, Rahul PR, Ganesh Kumar V, Banumathy R, Prathima PT, Raveendran M, Kumar KK & Balasubramanian P, 2009, *Interaction between sugarcane and* Colletotrichum falcatum *causing red rot: understanding disease resistance at transcription level*, Sugar Tech, **11**, pp. 44–50.

[224] Votaw DF & Orden A, 1952, *The personnel assignment problem*, Symposium on Linear Inequalities and Programming, SCOOP 10, US Air Force, pp. 155–163.

[225] Wang JB, 2008, *Single-machine scheduling with past-sequence-dependent setup times and time-dependent learning effect*, Computers & Industrial Engineering, **55**, pp. 584–591.

[226] Weintraub A, Epstein R, Morales R, Seron, J & Traverso, P, 1996, *A truck scheduling system improves efficiency in the forest industries*, Interfaces, **26:4**, pp. 1–12.

[227] Weintraub A & Murray AT, 2006, *Review of combinatorial problems induced by spatial forest harvesting planning*, Applied Mathematics, **154**, pp. 867–879.

[228] Wikipedia, 2010, *2000s energy crisis*, [Online], [Cited June 21st, 2010], Available from http://en.wikipedia.org/wiki/2000s_energy_crisis

[229] Wikipedia, 2010, *Muslim conquest of Persia*, [Online], [Cited June 22nd, 2010], Available from http://en.wikipedia.org/wiki/Muslim_conquest_of_Persia

[230] Wikipedia, 2010, *Sugarcane*, [Online], [Cited June 22nd, 2010], Available from http://en.wikipedia.org/wiki/Sugarcane

[231] Wikipedia, 2010, *Voyages of Christopher Columbus*, [Online], [Cited June 22nd, 2010], Available from http://en.wikipedia.org/wiki/Voyages_of_Christopher_Columbus#Second_voyage

[232] Wikipedia, 2010, *Wars of Alexander the Great*, [Online], [Cited June 22nd, 2010], Available from http://en.wikipedia.org/wiki/Wars_of_Alexander_the_Great

[233] Winston WL, 2004, *Operations Research Applications and Algorithms*, Brooks/Cole—Thomson Learning, Belmont (CA).

[234] Wood RA, 1973, *Varietal differences in rate of deterioration of whole stalk sugarcane*, Proceedings of the South African Sugar Technologists' Association, **47**, pp. 133–139.

[235] Wood RA, 1976, *Cane deterioration as affected by billet size, delay in milling and other factors*, Proceedings of the South African Sugar Technologists' Association, **50**, pp. 12–17.

[236] Wood RA, du Toit JL & Bruijn J, 1972, *Deterioration losses in whole stalk sugarcane*, Proceedings of the South African Sugar Technologists' Association, **46**, pp. 151–157.

[237] Xing LN, Chen YW, Yang KW, Hou F, Shen XS & Cai HP, 2008, *A hybrid approach combining an improved genetic algorithm and optimization strategies for the asymmetric traveling salesman problem*, Engineering Applications of Artificial Intelligence, **21**, pp. 1370–1380.

[238] Yagiura M, Ibaraki T & Glover F, 2004, *An ejection chain approach for the generalized assignment problem*, INFORMS Journal on Computing, **16**, pp. 133–151.

[239] Yagiura M, Ibaraki T & Glover F, 2006, *A path relinking approach with ejection chains for the generalized assignment problem*, European Journal of Operational Research, **169**, pp. 548–569.

APPENDIX A

Computer code

Contents

This appendix contains the commented computer code in the final DSS implementation of §8.2.

A.1 Visual Basic for Excel code for command button "Update Trends"

Below, the VBA code initiated by the command button "Update Trends" is shown, with comments following an apostrophe (').

```
Private Sub CommandButton3_Click() 'Sub-procedure initiated by clicking the "Update trends" button.
Dim NOV, NOE, HF, TNOF, NOF, FDY, LOS, LDS, FDS, SD, EP, NOP As Integer, RVP As Single
Dim counter, counter1 As Long
'Compute the number of varieties:
NOV = Application.WorksheetFunction.CountA(Worksheets("V").Range("A2:A999"))
'Compute the number of events:
NOE = Application.WorksheetFunction.CountA(Worksheets("EV").Range("A2:A999"))
'Compute the number of situations:
NOS = Application.WorksheetFunction.CountA(Worksheets("S").Range("A2:A999"))
'Compute the number of field aspects:
NOFD = Application.WorksheetFunction.CountA(Worksheets("FD").Range("A2:A999"))
'Capture the harvesting front name or number from the UI:
HF = Worksheets("Start").Range("Z22").Value
'Compute the total number of fields in database:
TNOF = Application.WorksheetFunction.CountA(Worksheets("Start").Range("B2:B999"))
'Compute the number of fields to be included in schedule:
NOF = Application.WorksheetFunction.CountIf(Worksheets("FDB").Range("A2:A999"), HF)
'Again account for harvested fields:
For counter1 = 1 To TNOF
    If Worksheets("FDB").Cells(counter1 + 1, 1).Value = HF Then
        RemainingFraction = 1
        If Worksheets("FDB").Cells(counter1 + 1, 19).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 21).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 22).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 24).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 25).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 27).Value / 100
        End If
        If RemainingFraction <= 0 Then
            NOF = NOF - 1
        End If
    End If
Next counter1
'Capture the first day of the year
FDY = Worksheets("Start").Range("Z2").Value
'Capture the number of periods in the entire year.
```

```
NOP = 52
Dim counter2, counter3, counter4 As Integer
'Define variables for base models and event models:
Dim RVij(), B0, B1, B11, BEC0, BEC1, RVBase, RVEV1, RVEV2, RVEV3, VCoeff, SitCoeff, FDCoeff As Single
'Dimension the RV matrix:
ReDim RVij(1 To TNOF, 1 To NOP)
'Compute RV values across all fields and periods accounting for all events:
For counter1 = 1 To TNOF 'Do this for all fields in the database:
    Sit = Worksheets("FDB").Cells(counter1 + 1, 4): FD = Worksheets("FDB").Cells(counter1 + 1, 5):
     V = Worksheets("FDB").Cells(counter1 + 1, 7)
    A0 = Worksheets("FDB").Cells(counter1 + 1, 12): A1 = Worksheets("FDB").Cells(counter1 + 1, 13):
    A11 = Worksheets("FDB").Cells(counter1 + 1, 14)
    B0 = Worksheets("FDB").Cells(counter1 + 1, 16): B1 = Worksheets("FDB").Cells(counter1 + 1, 17):
    B11 = Worksheets("FDB").Cells(counter1 + 1, 18)
    EV1 = Worksheets("FDB").Cells(counter1 + 1, 19): EV1D = Worksheets("FDB").Cells(counter1 + 1, 20):
    EV1P = Worksheets("FDB").Cells(counter1 + 1, 21)
    EV2 = Worksheets("FDB").Cells(counter1 + 1, 22): EV2D = Worksheets("FDB").Cells(counter1 + 1, 23):
    EV2P = Worksheets("FDB").Cells(counter1 + 1, 24)
    EV3 = Worksheets("FDB").Cells(counter1 + 1, 25): EV3D = Worksheets("FDB").Cells(counter1 + 1, 26):
    EV3P = Worksheets("FDB").Cells(counter1 + 1, 27)
    For counter2 = 1 To NOP 'Do this for all periods in the entire year:
        JD = counter2 * 7 'Compute the julian date of the last day of this period.
        RVBase = B0 + B1 * JD + B11 * JD ^ 2 'Compute base RV model value
        RVEV1 = 0: RVEV2 = 0: RVEV3 = 0 'Initialise effect values of the three events.
        If EV1D > FDY - 365 Then 'Do this if event 1 occurred before the first day of last year.
            If EV1D < JD + FDY Then 'Do if it occurred before the last day of this period.
                For counter3 = 1 To NOE 'Check all events in the events database.
                    If Worksheets("EV").Cells(counter3 + 1, 1) = EV1 Then 'Do this for the matching event:
                        'Let the current decrease coefficients for event 1 be those of the correct event in the database:
                        BEC0 = Worksheets("EV").Cells(counter3 + 1, 4): BEC1 = Worksheets("EV").Cells(counter3 + 1, 5)
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE 'Check all events across the varieties database:
                    If Worksheets("V").Cells(1, counter3 + 7) = EV1 Then 'Do this for the matching variety:
                        For counter4 = 1 To NOV
                            If Worksheets("V").Cells(counter4 + 1, 1) = V Then
                                'Let the current variety interaction coefficient be that of the matching variety and event:
                                VCoeff = Worksheets("V").Cells(counter4 + 1, counter3 + 7)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE 'Check all events across the toposequences database:
                    If Worksheets("S").Cells(1, counter3 + 1) = EV1 Then
                        For counter4 = 1 To NOS
                            If Worksheets("S").Cells(counter4 + 1, 1) = Sit Then
                                'Let the current locational interaction coefficient be that of the matching location and event:
                                SitCoeff = Worksheets("S").Cells(counter4 + 1, counter3 + 1)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE 'Check all events across the aspects database:
                    If Worksheets("FD").Cells(1, counter3 + 1) = EV1 Then
                        For counter4 = 1 To NOFD
                            If Worksheets("FD").Cells(counter4 + 1, 1) = FD Then
                                'Let the current locational interaction coefficient be that of the matching aspect and event:
                                FDCoeff = Worksheets("FD").Cells(counter4 + 1, counter3 + 1)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                'Compute the first event's alteration effect:
                RVEV1 = (BEC0 + BEC1 * (JD + FDY - EV1D)) * VCoeff * SitCoeff * FDCoeff
            End If
        End If
        'Do for event 2 what was done for event 1:
        If EV2D > FDY - 365 Then
            If EV2D < JD + FDY Then
                For counter3 = 1 To NOE
                    If Worksheets("EV").Cells(counter3 + 1, 1) = EV2 Then
                        BEC0 = Worksheets("EV").Cells(counter3 + 1, 4): BEC1 = Worksheets("EV").Cells(counter3 + 1, 5)
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE
                    If Worksheets("V").Cells(1, counter3 + 7) = EV2 Then
                        For counter4 = 1 To NOV
                            If Worksheets("V").Cells(counter4 + 1, 1) = V Then
                                VCoeff = Worksheets("V").Cells(counter4 + 1, counter3 + 7)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE
                    If Worksheets("S").Cells(1, counter3 + 1) = EV2 Then
```

```
                        For counter4 = 1 To NOS
                            If Worksheets("S").Cells(counter4 + 1, 1) = Sit Then
                                SitCoeff = Worksheets("S").Cells(counter4 + 1, counter3 + 1)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE
                    If Worksheets("FD").Cells(1, counter3 + 1) = EV2 Then
                        For counter4 = 1 To NOFD
                            If Worksheets("FD").Cells(counter4 + 1, 1) = FD Then
                                FDCoeff = Worksheets("FD").Cells(counter4 + 1, counter3 + 1)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                RVEV2 = (BEC0 + BEC1 * (JD + FDY - EV2D)) * VCoeff * SitCoeff * FDCoeff
            End If
        End If
        'Do for event 3 what was done for events 1 and 2:
        If EV3D > FDY - 365 Then
            If EV3D < JD + FDY Then
                For counter3 = 1 To NOE
                    If Worksheets("EV").Cells(counter3 + 1, 1) = EV3 Then
                        BEC0 = Worksheets("EV").Cells(counter3 + 1, 4): BEC1 = Worksheets("EV").Cells(counter3 + 1, 5)
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE
                    If Worksheets("V").Cells(1, counter3 + 7) = EV3 Then
                        For counter4 = 1 To NOV
                            If Worksheets("V").Cells(counter4 + 1, 1) = V Then
                                VCoeff = Worksheets("V").Cells(counter4 + 1, counter3 + 7)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE
                    If Worksheets("S").Cells(1, counter3 + 1) = EV3 Then
                        For counter4 = 1 To NOS
                            If Worksheets("S").Cells(counter4 + 1, 1) = Sit Then
                                SitCoeff = Worksheets("S").Cells(counter4 + 1, counter3 + 1)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE
                    If Worksheets("FD").Cells(1, counter3 + 1) = EV3 Then
                        For counter4 = 1 To NOFD
                            If Worksheets("FD").Cells(counter4 + 1, 1) = FD Then
                                FDCoeff = Worksheets("FD").Cells(counter4 + 1, counter3 + 1)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                RVEV3 = (BEC0 + BEC1 * (JD + FDY - EV3D)) * VCoeff * SitCoeff * FDCoeff
            End If
        End If
        'Ensure that RV values are nonnegative:
        If RVBase + RVEV1 + RVEV2 + RVEV3 < 0 Then
            RVij(counter1, counter2) = 0
        Else
            RVij(counter1, counter2) = RVBase + RVEV1 + RVEV2 + RVEV3
        End If
    Next counter2
Next counter1
'Clear the RV part of the database and paste the new RV matrix:
Worksheets("FDB").Range(Worksheets("FDB").Cells(2, 31), Worksheets("FDB").Cells(999, NOP + 30)).ClearContents
Worksheets("FDB").Range(Worksheets("FDB").Cells(2, 31), Worksheets("FDB").Cells(TNOF + 1, NOP + 30)).Value = RVij

'Do the yield matrix in the same fashion as the RV matrix.
Dim Yij(), YBase As Single
ReDim Yij(1 To TNOF, 1 To NOP)

For counter1 = 1 To TNOF
    MAV = Worksheets("Start").Cells(counter1 + 1, 23): LHD = Worksheets("FDB").Cells(counter1 + 1, 9)
    Sit = Worksheets("FDB").Cells(counter1 + 1, 4): FD = Worksheets("FDB").Cells(counter1 + 1, 5):
    V = Worksheets("FDB").Cells(counter1 + 1, 7)
    A0 = Worksheets("FDB").Cells(counter1 + 1, 12): A1 = Worksheets("FDB").Cells(counter1 + 1, 13):
    A11 = Worksheets("FDB").Cells(counter1 + 1, 14)
    EV1 = Worksheets("FDB").Cells(counter1 + 1, 19): EV1D = Worksheets("FDB").Cells(counter1 + 1, 20):
    EV1P = Worksheets("FDB").Cells(counter1 + 1, 21)
    EV2 = Worksheets("FDB").Cells(counter1 + 1, 22): EV2D = Worksheets("FDB").Cells(counter1 + 1, 23):
    EV2P = Worksheets("FDB").Cells(counter1 + 1, 24)
    EV3 = Worksheets("FDB").Cells(counter1 + 1, 25): EV3D = Worksheets("FDB").Cells(counter1 + 1, 26):
```

```
EV3P = Worksheets("FDB").Cells(counter1 + 1, 27)

For counter2 = 1 To NOP
    JD = counter2 * 7 'Compute the Julian date of the last day of the period.
    Age = JD - LHD + FDY
    'Compensate age for periods of slow growth:
    If JD > 151 Then
        If JD > 243 Then
            LGT1 = 92
        Else
            LGT1 = JD - 151
        End If
    Else
        LGT1 = 0
    End If
    LJD = 365 - (FDY - LHD)
    If LJD < 243 Then
        If LJD < 151 Then
            LGT2 = 92
        Else
            LGT2 = 243 - LJD
        End If
    Else
        LGT2 = 0
    End If
    'Compute EGT:
    EGT = (Age - LGT1 - LGT2) / 30.4375 'Compute EGT by subtraction lost growth time from age.
    YBase = A0 + A1 * EGT + A11 * EGT ^ 2
    YEV1 = 0: YEV2 = 0: YEV3 = 0
    'Do as was done for RV:
    If EV1D > FDY - 365 Then
        If EV1D < JD + FDY Then
            For counter3 = 1 To NOE
                If Worksheets("EV").Cells(counter3 + 1, 1) = EV1 Then
                    AEC0 = Worksheets("EV").Cells(counter3 + 1, 2): AEC1 = Worksheets("EV").Cells(counter3 + 1, 3)
                    Exit For
                End If
            Next counter3
            For counter3 = 1 To NOE
                If Worksheets("V").Cells(1, counter3 + 7) = EV1 Then
                    For counter4 = 1 To NOV
                        If Worksheets("V").Cells(counter4 + 1, 1) = V Then
                            VCoeff = Worksheets("V").Cells(counter4 + 1, counter3 + 7)
                            Exit For
                        End If
                    Next counter4
                    Exit For
                End If
            Next counter3
            For counter3 = 1 To NOE
                If Worksheets("S").Cells(1, counter3 + 1) = EV1 Then
                    For counter4 = 1 To NOS
                        If Worksheets("S").Cells(counter4 + 1, 1) = Sit Then
                            SitCoeff = Worksheets("S").Cells(counter4 + 1, counter3 + 1)
                            Exit For
                        End If
                    Next counter4
                    Exit For
                End If
            Next counter3
            For counter3 = 1 To NOE
                If Worksheets("FD").Cells(1, counter3 + 1) = EV1 Then
                    For counter4 = 1 To NOFD
                        If Worksheets("FD").Cells(counter4 + 1, 1) = FD Then
                            FDCoeff = Worksheets("FD").Cells(counter4 + 1, counter3 + 1)
                            Exit For
                        End If
                    Next counter4
                    Exit For
                End If
            Next counter3
            YEV1 = (AEC0 + AEC1 * (JD - EV1D + FDY)) * VCoeff * SitCoeff * FDCoeff
       End If
    End If
    If EV2D > FDY - 365 Then
        If EV2D < JD + FDY Then
            For counter3 = 1 To NOE
                If Worksheets("EV").Cells(counter3 + 1, 1) = EV2 Then
                    AEC0 = Worksheets("EV").Cells(counter3 + 1, 2): AEC1 = Worksheets("EV").Cells(counter3 + 1, 3)
                    Exit For
                End If
            Next counter3
            For counter3 = 1 To NOE
                If Worksheets("V").Cells(1, counter3 + 7) = EV2 Then
                    For counter4 = 1 To NOV
                        If Worksheets("V").Cells(counter4 + 1, 1) = V Then
                            VCoeff = Worksheets("V").Cells(counter4 + 1, counter3 + 7)
                            Exit For
                        End If
                    Next counter4
                    Exit For
                End If
            Next counter3
            For counter3 = 1 To NOE
                If Worksheets("S").Cells(1, counter3 + 1) = EV2 Then
```

```
                        For counter4 = 1 To NOS
                            If Worksheets("S").Cells(counter4 + 1, 1) = Sit Then
                                SitCoeff = Worksheets("S").Cells(counter4 + 1, counter3 + 1)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE
                    If Worksheets("FD").Cells(1, counter3 + 1) = EV2 Then
                        For counter4 = 1 To NOFD
                            If Worksheets("FD").Cells(counter4 + 1, 1) = FD Then
                                FDCoeff = Worksheets("FD").Cells(counter4 + 1, counter3 + 1)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                YEV2 = (AEC0 + AEC1 * (JD - EV2D + FDY)) * VCoeff * SitCoeff * FDCoeff
            End If
        End If
        If EV3D > FDY - 365 Then
            If EV3D < JD + FDY Then
                For counter3 = 1 To NOE
                    If Worksheets("EV").Cells(counter3 + 1, 1) = EV3 Then
                        AEC0 = Worksheets("EV").Cells(counter3 + 1, 2): AEC1 = Worksheets("EV").Cells(counter3 + 1, 3)
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE
                    If Worksheets("V").Cells(1, counter3 + 7) = EV3 Then
                        For counter4 = 1 To NOV
                            If Worksheets("V").Cells(counter4 + 1, 1) = V Then
                                VCoeff = Worksheets("V").Cells(counter4 + 1, counter3 + 7)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE
                    If Worksheets("S").Cells(1, counter3 + 1) = EV3 Then
                        For counter4 = 1 To NOS
                            If Worksheets("S").Cells(counter4 + 1, 1) = Sit Then
                                SitCoeff = Worksheets("S").Cells(counter4 + 1, counter3 + 1)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                For counter3 = 1 To NOE
                    If Worksheets("FD").Cells(1, counter3 + 1) = EV3 Then
                        For counter4 = 1 To NOFD
                            If Worksheets("FD").Cells(counter4 + 1, 1) = FD Then
                                FDCoeff = Worksheets("FD").Cells(counter4 + 1, counter3 + 1)
                                Exit For
                            End If
                        Next counter4
                        Exit For
                    End If
                Next counter3
                YEV3 = (AEC0 + AEC1 * (JD - EV3D + FDY)) * VCoeff * SitCoeff * FDCoeff
            End If
        End If
        'In addition to how it was done for RV, add the manual adjustment values from the UI (MAV):
        If YBase + YEV1 + YEV2 + YEV3 + MAV < 0 Then
            Yij(counter1, counter2) = 0
        Else
            Yij(counter1, counter2) = YBase + YEV1 + YEV2 + YEV3 + MAV
        End If
    Next counter2
Next counter1
Worksheets("FDB").Range(Worksheets("FDB").Cells(2, 87), Worksheets("FDB").Cells(999, NOP + 86)).ClearContents
Worksheets("FDB").Range(Worksheets("FDB").Cells(2, 87), Worksheets("FDB").Cells(TNOF + 1, NOP + 86)).Value = Yij
'Compute deviations between cane yield trends and user estimates:
Dim Period As Integer
Dim DET() As Single
ReDim DET(1 To TNOF, 1)
For counter1 = 1 To TNOF
    ESTD = Worksheets("FDB").Cells(counter1 + 1, 10)
    JD = ESTD - FDY
    ESTY = Worksheets("FDB").Cells(counter1 + 1, 11).Value
    Period = PeriodFunction(JD)
    DET(counter1, 1) = ESTY - Yij(counter1, Period)
Next counter1
'Paste the deviations into the UI:
Worksheets("Start").Range(Worksheets("Start").Cells(2, 22), Worksheets("Start").Cells(999, 22)).ClearContents
Worksheets("Start").Range(Worksheets("Start").Cells(2, 22), Worksheets("Start").Cells(TNOF + 1, 22)).Value = DET

'Build the Cij matrix similarly as the RV and yield matrices:
HBaseCost = Worksheets("Start").Range("Z12").Value
ZBaseCost = Worksheets("Start").Range("Z13").Value
```

```
EventPenalty = Worksheets("Start").Range("Z16").Value
Dim Cij() As Single
ReDim Cij(1 To TNOF, 1 To NOP)
For counter = 1 To TNOF 'Do for all fields in the problem:
    Sit = Worksheets("FDB").Cells(counter + 1, 4): FD = Worksheets("FDB").Cells(counter + 1, 5):
    AC = Worksheets("FDB").Cells(counter + 1, 6)
    EV1 = Worksheets("FDB").Cells(counter + 1, 19): EV1D = Worksheets("FDB").Cells(counter + 1, 20):
    EV1P = Worksheets("FDB").Cells(counter + 1, 21) / 100
    EV2 = Worksheets("FDB").Cells(counter + 1, 22): EV2D = Worksheets("FDB").Cells(counter + 1, 23):
    EV2P = Worksheets("FDB").Cells(counter + 1, 24) / 100
    EV3 = Worksheets("FDB").Cells(counter + 1, 25): EV3D = Worksheets("FDB").Cells(counter + 1, 26):
    EV3P = Worksheets("FDB").Cells(counter + 1, 27) / 100
    HCost = HBaseCost

    For counter1 = 1 To NOP 'Do for all periods:
        If AC = 2 Then 'Check whether the field is an access category 2 field.
            ZCost = ZBaseCost 'In that case give it a zoneloading cost of 100 %.
        Else 'Otherwise:
            JD = counter1 * 7 'Compute the Julian date of the last day of the period.
            For counter2 = 1 To NOS
                If Worksheets("S").Cells(counter2 + 1, 1) = Sit Then
                    SitCoeff = Worksheets("S").Cells(counter2 + 1, 25)
                    Exit For
                End If
            Next counter2
            For counter2 = 1 To NOFD
                If Worksheets("FD").Cells(counter4 + 1, 1) = FD Then
                    FDCoeff = Worksheets("FD").Cells(counter4 + 1, 25)
                    Exit For
                End If
            Next counter2
            For counter2 = 1 To NOP
                If Worksheets("RW").Cells(1, counter2 + 1) = JD Then
                    WCP = Worksheets("RW").Cells(7, counter2 + 1) 'Capture wet conditions probability.
                    Exit For
                End If
            Next counter2
            'Add the wet conditions probability-based zone loading cost:
            ZCost = ZBaseCost * SitCoeff * FDCoeff * WCP
        End If
        JD = counter1 * 7: EV1Cost = 0
        'Capture relevant harvesting time windows for event 1:
        For counter2 = 1 To NOE
            If Worksheets("EV").Cells(counter2 + 1, 1) = EV1 Then
                HTW = Worksheets("EV").Cells(counter2 + 1, 6)
                HTD = Worksheets("EV").Cells(counter2 + 1, 7)
                EHD = Worksheets("EV").Cells(counter2 + 1, 8)
                Exit For
            End If
        Next counter2
        'Compute a penalty for harvesting later than HTW days after the event date:
        If JD + FDY > EV1D + HTW Then EV1Cost = EV1Cost + EV1P * EventPenalty * (JD + FDY - EV1D - HTW)
        'Compute a penalty for harvesting after HTD:
        If JD > HTD Then EV1Cost = EV1Cost + EV1P * EventPenalty * (JD - HTD)
        'Compute a penalty for harvesting before EHD days has elapsed after the event date.
        If JD + FDY < EV1D + EHD Then EV1Cost = EV1Cost + EV1P * EventPenalty * (EV1D + EHD - JD - FDY)
        'Do the same for event 2:
        EV2Cost = 0
        For counter2 = 1 To NOE
            If Worksheets("EV").Cells(counter2 + 1, 1) = EV2 Then
                HTW = Worksheets("EV").Cells(counter2 + 1, 6)
                HTD = Worksheets("EV").Cells(counter2 + 1, 7)
                EHD = Worksheets("EV").Cells(counter2 + 1, 8)
                Exit For
            End If
        Next counter2
        If JD + FDY > EV2D + HTW Then EV2Cost = EV2Cost + EV2P * EventPenalty * (JD + FDY - EV2D - HTW)
        If JD > HTD Then EV2Cost = EV2Cost + EV2P * EventPenalty * (JD - HTD)
        If JD + FDY < EV2D + EHD Then EV2Cost = EV2Cost + EV2P * EventPenalty * (EV2D + EHD - JD - FDY)

        'Do the same for event 3:
        EV3Cost = 0
        For counter2 = 1 To NOE
            If Worksheets("EV").Cells(counter2 + 1, 1) = EV3 Then
                HTW = Worksheets("EV").Cells(counter2 + 1, 6)
                HTD = Worksheets("EV").Cells(counter2 + 1, 7)
                EHD = Worksheets("EV").Cells(counter2 + 1, 8)
                Exit For
            End If
        Next counter2
        If JD + FDY > EV3D + HTW Then EV3Cost = EV3Cost + EV3P * EventPenalty * (JD + FDY - EV3D - HTW)
        If JD > HTD Then EV3Cost = EV3Cost + EV3P * EventPenalty * (JD - HTD)
        If JD + FDY < EV3D + EHD Then EV3Cost = EV3Cost + EV3P * EventPenalty * (EV3D + EHD - JD - FDY)
        'Sum all costs for the field and the period and enter into the cost matrix.
        Cij(counter, counter1) = HBaseCost + ZCost + EV1Cost + EV2Cost + EV3Cost
    Next counter1
Next counter
'Paste the cost matrix into the field database:
Worksheets("FDB").Range(Worksheets("FDB").Cells(2, 141), Worksheets("FDB").Cells(999, NOP + 140)).ClearContents
Worksheets("FDB").Range(Worksheets("FDB").Cells(2, 141), Worksheets("FDB").Cells(TNOF + 1, NOP + 140)).Value = Cij

'Build Mill RV matrix
Dim MRVij() As Single
ReDim MRVij(1 To TNOF, 1 To NOP)
'Capture the mill trend name from the UI:
```

```
Mill = Worksheets("Start").Range("Z9").Value
'The mill trend coefficients are located in the varieties worksheet "V":
For counter2 = 1 To NOV 'Check all entries in the "V" worksheet.
    If Worksheets("V").Cells(counter2 + 1, 1) = Mill Then 'Find the matching mill trend in worksheet "V".
        MBC0 = Worksheets("V").Cells(counter2 + 1, 5):
        MBC1 = Worksheets("V").Cells(counter2 + 1, 6):
        MBC11 = Worksheets("V").Cells(counter2 + 1, 7)
        Exit For
    End If
Next counter2
'Populate the Mill RV matrix:
For counter = 1 To TNOF
    For counter1 = 1 To NOP
        JD = counter1 * 7
        MRVij(counter, counter1) = MBC0 + MBC1 * JD + MBC11 * JD ^ 2
    Next counter1
Next counter
Worksheets("FDB").Range(Worksheets("FDB").Cells(2, 195), Worksheets("FDB").Cells(999, NOP + 194)).ClearContents
Worksheets("FDB").Range(Worksheets("FDB").Cells(2, 195), Worksheets("FDB").Cells(TNOF + 1, NOP + 194)).Value = MRVij

'Compute the relative RV matrix values:
Dim SumMRV, AverageMRV(), RRVij() As Single
ReDim RRVij(1 To TNOF, 1 To NOP), AverageMRV(1 To TNOF)
For counter = 1 To TNOF
    SumMRV = 0
    For counter1 = 1 To NOP
        MRV = MRVij(counter, counter1)
        SumMRV = SumMRV + MRV
    Next counter1
    AverageMRV(counter) = SumMRV / NOP
Next counter
For counter = 1 To TNOF
    For counter1 = 1 To NOP
        RRVij(counter, counter1) = RVij(counter, counter1) - MRVij(counter, counter1) + AverageMRV(counter)
    Next counter1
Next counter
'Paste the relative RV matrix into the field database:
Worksheets("FDB").Range(Worksheets("FDB").Cells(2, 249), Worksheets("FDB").Cells(999, NOP + 248)).ClearContents
Worksheets("FDB").Range(Worksheets("FDB").Cells(2, 249), Worksheets("FDB").Cells(TNOF + 1, NOP + 248)).Value = RRVij
End Sub

Public Function PeriodFunction(JD) 'This function returns the week number given a Julian date.
    Dim P As Integer
    P = 1 + (JD - 4) / 7
    If P > 52 Then P = 52

    PeriodFunction = P
End Function
```

A.2 Visual Basic for Excel code for command button "Schedule!"

```
Option Base 1

Private Sub CommandButton1_Click() 'Sub-procedure initiated by clicking the "Schedule!" button.
Dim HF, TNOF, NOF, LOS, LDS, FDS, SD, EP, NOP As Integer, RVP As Single 'Initialise variables.
Worksheets("Mij").Cells.ClearContents 'Clear space for Mij matrix.
Worksheets("Pij").Cells.ClearContents 'Clear space for Pij matrix.
HF = Worksheets("Start").Range("Z22").Value 'Capture harvesting front name or number.
'Compute total number of fields in database:
TNOF = Application.WorksheetFunction.CountA(Worksheets("Start").Range("B2:B999"))
'Compute number of fields to be included in schedule:
NOF = Application.WorksheetFunction.CountIf(Worksheets("FDB").Range("A2:A999"), HF)
'Discount harvested fields from NOF.
For counter1 = 1 To TNOF 'Do for all fields in database.
    If Worksheets("FDB").Cells(counter1 + 1, 1).Value = HF Then 'Do only if field is in correct harvesting front.
        RemainingFraction = 1
        If Worksheets("FDB").Cells(counter1 + 1, 19).Value = "Harvest" Then 'Check if first event is a harvest event.
            'Remove the fraction having been harvested:
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 21).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 22).Value = "Harvest" Then 'Check if second event is a harvest event.
            'Remove the fraction having been harvested:
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 24).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 25).Value = "Harvest" Then 'Check if third event is a harvest event.
            'Remove the fraction having been harvested.
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 27).Value / 100
        End If
        If RemainingFraction <= 0 Then 'If field is completely harvested, discount it from NOF.
            NOF = NOF - 1
        End If
    End If
Next counter1
LOS = Worksheets("Start").Range("Z21").Value - Worksheets("Start").Range("Z19").Value 'Compute length of schedule in days
FDS = Worksheets("Start").Range("Z19").Value - Worksheets("Start").Range("Z2").Value 'Compute first Julian day of schedule
LDS = Worksheets("Start").Range("Z21").Value - Worksheets("Start").Range("Z2").Value 'Compute last Julian day of schedule
SP = PeriodFunction(FDS) 'Compute starting period of the schedule.
EP = PeriodFunction(LDS) 'Compute ending period of the schedule.
NOP = EP - SP + 1 'Compute the number of periods covered by the schedule.
```

```
RVP = Worksheets("Start").Range("Z8").Value 'Capture RV-price from UI.
'Mij and Pij matrices:
counter2 = 0
Dim MijMatrix() 'Define a matrix containing field by period cane yield.
Dim PijMatrix() 'Define a matrix containing field by period profits.
Dim CTAij() As Single   'Define a vector containing field by period cutting times based on hectares.
Dim FieldNamesVector() As Variant   'Define a vector containing the names of the fields.
Dim TotArea, Area, TotYield, Mass As Single 'Define variables for areas and yields.
ReDim FieldNamesVector(1 To NOF) 'Define vector for field names.
ReDim MijMatrix(1 To NOF, 1 To NOP) 'Dimension yield matrix.
ReDim PijMatrix(1 To NOF, 1 To NOP) 'Dimension profit matrix.
ReDim CTAij(1 To NOF) 'Dimension cutting times vector
'Remove harvested fields and build problem-specific vectors and matrices.
For counter1 = 1 To TNOF
    Mass = 0
    If Worksheets("FDB").Cells(counter1 + 1, 1).Value = HF Then 'Check harvesting front.
        RemainingFraction = 1
        If Worksheets("FDB").Cells(counter1 + 1, 19).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 21).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 22).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 24).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 25).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 27).Value / 100
        End If
        If RemainingFraction > 0 Then   'Do this if there is anything left of the field.
            counter2 = counter2 + 1
            For counter3 = 1 To NOP
                'Adding field name to vector:
                FieldNamesVector(counter2) = Worksheets("FDB").Cells(counter1 + 1, 2).Value
                'Capture its area from the field database:
                Area = Worksheets("FDB").Cells(counter1 + 1, 3).Value * RemainingFraction
                'Capture the cane yield per hectare from the field database:
                Yield = Worksheets("FDB").Cells(counter1 + 1, 85 + SP + counter3).Value
                'Capture the relative RV from the field database:
                RRV = Worksheets("FDB").Cells(counter1 + 1, 247 + SP + counter3).Value / 100
                'Capture harvesting cost from the field database:
                Cost = Worksheets("FDB").Cells(counter1 + 1, 139 + SP + counter3).Value
                'Compute the cane yield and add to total cane yield for the field:
                Mass = Mass + Area * Yield
                'Compute cane yield:
                MijMatrix(counter2, counter3) = Area * Yield
                'Compute profit:
                PijMatrix(counter2, counter3) = Area * Yield * (RVP * RRV - Cost)
            Next counter3
            'Add area to total area:
            TotArea = TotArea + Area
            'Compute the sum of all field cane yield averages:
            TotYield = TotYield + Mass / NOP
        End If
    End If
Next counter1
'Paste Mij and Pij matrices into a worksheet each:
Worksheets("Mij").Range(Worksheets("Mij").Cells(1, 1), Worksheets("Mij").Cells(NOF, NOP)).Value = MijMatrix
Worksheets("Pij").Range(Worksheets("Pij").Cells(1, 1), Worksheets("Pij").Cells(NOF, NOP)).Value = PijMatrix
'Prepare the worksheet that will contain the final harvesting sequence.
Worksheets("Sequence").Cells.ClearContents

'Cutting times vector CTAij
counter2 = 0
'This again accounts for harvested fields:
For counter1 = 1 To TNOF
    If Worksheets("FDB").Cells(counter1 + 1, 1).Value = HF Then
        RemainingFraction = 1
        If Worksheets("FDB").Cells(counter1 + 1, 19).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 21).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 22).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 24).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 25).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 27).Value / 100
        End If
        If RemainingFraction > 0 Then
            counter2 = counter2 + 1
            Area = Worksheets("FDB").Cells(counter1 + 1, 3).Value * RemainingFraction
            'Compute the cutting time for each field (counter2):
            CTAij(counter2) = Application.WorksheetFunction.RoundDown(Area / TotArea * LOS, 3)
        End If
    End If
Next counter1
'Set the cutting times matrix.
CTij = CTAij

'The Nuij matrix is used in the ejection chain function.
'Compute the matrix of relative profits by field and harvest period.
Dim Nuij()
ReDim Nuij(1 To NOF, 1 To NOP)
For i = 1 To NOF
    PiMax = -10 ^ 30
    For j = 1 To NOP
        If PijMatrix(i, j) > PiMax Then PiMax = PijMatrix(i, j)
    Next j
    For j = 1 To NOP
```

```
        Nuij(i, j) = PijMatrix(i, j) / PiMax
    Next j
Next i

'DRD calculations
Dim DRDByAverageYield, MaxFieldYield, DRDByMaxYield, MaxFieldProfit, MaxProfit As Double
'Compute the average yield across all fields and periods in the problem:
DRDByAverageYield = Application.WorksheetFunction.Average(Worksheets("Mij").
Range(Worksheets("Mij").Cells(1, 1), Worksheets("Mij").Cells(NOF, NOP)).Value) * NOF / LOS
For counter1 = 1 To NOF 'Do this for all fields in the problem.
    'Find the maximum yield for the field:
    MaxFieldYield = Application.WorksheetFunction.Max(Worksheets("Mij").Range(Worksheets("Mij").
Cells(counter1, 1), Worksheets("Mij").Cells(counter1, NOP)).Value)
    'Add to the total:
    DRDByMaxYield = DRDByMaxYield + MaxFieldYield
Next counter1
DRDByMaxYield = DRDByMaxYield / LOS 'Compute an average yield per day.
'Paste the DRD based on the average into the UI:
Worksheets("Start").Range("Z23").Value = DRDByAverageYield
'Paste the DRD based on the maximum yields into the UI:
Worksheets("Start").Range("Z24").Value = DRDByMaxYield
'Compute an upper bound on the objective value
For counter1 = 1 To NOF 'Do this for all fields in the problem.
    'Compute the maximum profit for the field:
    MaxFieldProfit = Application.WorksheetFunction.Max(Worksheets("Pij").Range(Worksheets("Pij").
Cells(counter1, 1), Worksheets("Pij").Cells(counter1, NOP)).Value)
    'Add the profit to the total:
    MaxProfit = MaxProfit + MaxFieldProfit
Next counter1
'Paste the upper bound into the UI:
Worksheets("Start").Range("Z25").Value = MaxProfit
Dim TimeLimit As Variant
'Capture the TimeLimit variable from the UI:
newHour = Hour(Now()) + Worksheets("Start").Range("Z28"):
newMinute = Minute(Now()) + Worksheets("Start").Range("Z29"):
newSecond = Second(Now()) + Worksheets("Start").Range("Z30")
EndTime = TimeSerial(newHour, newMinute, newSecond)
'x(j) is the solution representation, where x (j) = i corresponds to field i having sequence number j.
'Generate a random permutaton of a range of values by applying the RndSubset function:
x = RndSubset(NOF) 'Do this for the number of fields in the problem.
Worksheets("Start").Range("Z26") = OV(x, NOF, NOP, PijMatrix, CTij) 'Compute the objective function value (OFV).
'Apply the tabu search function:
xfinal = TabuSearch(x, NOF, NOP, SP, MijMatrix, PijMatrix, CTij, Nuij, EndTime)
'Paste the OFB into the UI:
Worksheets("Start").Range("Z26") = OV(xfinal, NOF, NOP, PijMatrix, CTij)
'Paste the solution vector into the UI:
Worksheets("Sequence").Range(Worksheets("Sequence").Cells(1, 1), Worksheets("Sequence").Cells(1, NOF)).Value = xfinal
'Paste the field names vector into the UI.
Worksheets("Sequence").Range(Worksheets("Sequence").Cells(2, 1), Worksheets("Sequence").Cells(2, NOF)).Value = FieldNamesVector
End Sub


Public Function RndSubset(ByVal Length) As Variant

'This function returns a pseudo-random permuation of a sequence of elements.
'Generate a sequence of random numbers with length Length.
Dim RndRanks() As Single
ReDim RndRanks(1 To Length, 1 To 2)
Randomize
For fcounter1 = 1 To Length
    RndRanks(fcounter1, 1) = fcounter1
    RndRanks(fcounter1, 2) = Rnd
Next fcounter1
Dim fcounter As Long
Dim Ordering() As Long
ReDim Ordering(1 To Length)

For fcounter = 1 To Length
    MaximumValue = 0
    For fcounter1 = 1 To Length
        If RndRanks(fcounter1, 2) >= MaximumValue Then
            Largest = fcounter1:
            MaximumValue = RndRanks(fcounter1, 2)
        End If
    Next fcounter1
    Ordering(fcounter) = Largest
    RndRanks(Largest, 2) = -1
Next fcounter
RndSubset = Ordering
End Function

Public Function OV(x, NOF, NOP, PijMatrix, CTij) As Variant
'This function generates the objective value given a sequence.
'Define time of harvest variable (TOH) and value variable (V). Unit is day of the year.
'Define harvesting period.
Dim P As Integer
Dim V, CTM, TOH As Single
Dim Yield, Val, FNo, fc As Long
TOH = 1
For fc = 1 To NOF
    FNo = x(fc)
    P = 1 + (TOH - 4) / 7
    If P > 52 Then P = 52
    Val = PijMatrix(FNo, P)
    CTM = CTij(FNo)
    TOH = TOH + CTM
    V = V + Val
Next fc
OV = V
End Function
Public Function CT(x, NOF, NOP, SP, CTij) As Variant
'This function generates the Julian date of the first cutting day of each field in x.
Dim CuttingJD(), HP As Integer
ReDim CuttingJD(1 To NOF)
Dim CTM, TOH As Single
Dim FNo, fcounter As Long
TOH = Worksheets("Start").Range("Z19").Value - Worksheets("Start").Range("Z2").Value - (SP - 1) * 7 'CTij is defined
'only on the periods of the schedule - not all 52 periods - so 7 *the number of preceeding periods must be deducted.
For fcounter = 1 To NOF
    CuttingJD(fcounter) = TOH + 7 * (SP - 1)
    FNo = x(fcounter)
    HP = PeriodFunction(TOH)
    CTM = CTij(FNo)
    TOH = TOH + CTM
Next fcounter
CT = CuttingJD
End Function

Public Function Attribute1(S, x, NOF) As Variant
'This attribute returns the 1st dropped neighbouring pair (the left neighbouring pair).
Dim Edge(1 To 2) 'Give a neighbouring pair (np) two elements.
If S(1) = 1 Then 'Do if the first element of the current solution is the moving field:
    'Set the first element of the first dropped np to be the last field in the solution:
    Edge(1) = x(NOF):
    'Set the second element of the first dropped np to be the first field in the solution:
    Edge(2) = x(1)
Else
    'Set the first element of the first dropped np to be the field that precedes the moving field in the solution:
    Edge(1) = x(S(1) - 1):
    'Set the second element of the first dropped np to be the moving field:
    Edge(2) = x(S(1))
End If
Attribute1 = Edge
End Function
Public Function Attribute2(S, x, NOF) As Variant
'This attribute returns the 2nd dropped neighbouring pair (the right neighbouring pair).
Dim Edge(1 To 2)
If S(1) = NOF Then 'Do if the last element of the current solution is the moving field:
    'Set the first element of the second dropped np to be the moving field:
    Edge(1) = NOF:
    'Set the second element of the second dropped np to be the first field in the solution:
    Edge(2) = x(1)
Else
    'Set the first element of the second dropped np to be the moving field:
    Edge(1) = x(S(1)):
    'Set the second element of the second dropped np to be the field that succeeds the moving field in the solution:
    Edge(2) = x(S(1) + 1)
End If
Attribute2 = Edge
End Function
Public Function Attribute3(S, x, NOF) As Variant
'This attribute returns the 3rd dropped neighbouring pair (the split neighbouring pair).
Dim Edge(1 To 2)
If S(2) = NOF Then 'Do if the moving field is moving into the last position in the solution:
    'Set the first element of the split np to be the last field in the solution:
    Edge(1) = x(NOF):
    'Set the second element of the split np to be the first field in the solution:
    Edge(2) = x(1)
ElseIf S(2) = 1 Then 'Do if the moving field is moving to the first position in the solution:
    'Set the first element of the split np to be the last field of the solution:
    Edge(1) = x(NOF):
    'Set the second element of teh split np to be the first element of the solution:
    Edge(2) = x(1)
ElseIf S(1) < S(2) Then 'Do if the field is moving to a later position in the solution:
    'Set the first edge of the split np to be the field in the target position of the moving field:
    Edge(1) = x(S(2)):
    'Set the second edge of the split np to be the field succeeding
    'the field in the target position of the moving field:
    Edge(2) = x(S(2) + 1)
Else
    'Set the first edge of the split np to be the field preceeding
    'the field in the target position of the moving field:
    Edge(1) = x(S(2) - 1)
    'Set the second edge of the split np to be the field in the target position of the moving field:
    Edge(2) = x(S(2))
End If
Attribute3 = Edge
End Function
Public Function Attribute4(S, x, NOF) As Variant
'This attribute returns the 1st added neighbouring pair.
Dim Edge(1 To 2)
If S(2) = 1 Then 'Do if the target position of the moving field is the first position of the solution:
    If S(1) = NOF Then 'Do if the moving field is moving from the last position in the solution:
        'Set the first element of the first added np to be the field preceeding the moving field:
        Edge(1) = x(S(1) - 1)
        'Set the second element of the first added np to be the last field in the solution:
        Edge(2) = x(NOF)
    Else
        'Set the first element of the first added np to be the last field in the solution:
        Edge(1) = x(NOF):
```

```
        'Set the second element of the first added np to be the moving field.
        Edge(2) = x(S(1))
    End If
ElseIf S(1) < S(2) Then 'Do if the moving field is moving to a later part of the solution:
    'Set the first element of the first added np to be the field in the moving field's target position:
    Edge(1) = x(S(2)):
    'Set the second element of the first added np to be the moving field:
    Edge(2) = x(S(1))
Else
    'Set the first element of the first added np to be the field preceeding
    'the field in the moving field's target position:
    Edge(1) = x(S(2) - 1):
    'Set the second element to be the moving field:
    Edge(2) = x(S(1))
End If
Attribute4 = Edge
End Function
Public Function Attribute5(S, x, NOF) As Variant
'This attribute returns the 2nd added neighbouring pair.
Dim Edge(1 To 2)
If S(2) = NOF Then 'Do if the moving field is moving to the last position in the solution:
    If S(1) = 1 Then 'Do if the moving field is moving from the first position in the solution:
        'Set the first element of the second added np to be the moving field:
        Edge(1) = x(1)
        'Set the second element of the second added np to be the field in the second position:
        Edge(2) = x(2)
    Else
        'Set the first element of the second added np to be the moving field:
        Edge(1) = x(S(1))
        'Set the second element of the second added np to be the field
        'in the first position of the solution:
        Edge(2) = x(1)
    End If
ElseIf S(1) < S(2) Then 'Do if the moving field is moving to a later position in the solution:
    'Set the first element of the second added np to be the moving field:
    Edge(1) = x(S(1)):
    'Set the second element of the second added np to be the field succeding
    'the field in the target position of the moving field:
    Edge(2) = x(S(2) + 1)
Else
    'Set the first element of the second added np to be the moving field:
    Edge(1) = x(S(1)):
    'Set the second element of the second added np to be the field in the target position:
    Edge(2) = x(S(2))
End If
Attribute5 = Edge
End Function
Public Function Attribute6(S, x, NOF) As Variant
'This attribute returns the 3rd added edge.
Dim Edge(1 To 2)
If S(1) = 1 Then 'Do if the moving field is in the first position of the solution:
    If S(2) = NOF Then
        'Set the first element of the third added np to be the moving field:
        Edge(1) = x(1):
        'Set the second element of the third added np to be the field in
        'the target position of the moving field:
        Edge(2) = x(2)
    Else
        'Set the first element of the third added np to be the field in the
        'last position of the solution:
        Edge(1) = x(NOF):
        'Set the second element of the third added np to be the field in the second
        'position of the solution:
        Edge(2) = x(2)
    End If
ElseIf S(1) = NOF Then 'Do if the moving field is in the last position of the solution:
    If S(2) = 1 Then 'Do if the moving field is moving to the first position in the solution:
        'Set the first element of the third added np to be the second-to-last field in the solution:
        Edge(1) = x(NOF - 1)
        'Set the second element of the third added np to be the last field in the solution:
        Edge(2) = x(NOF)
    Else
        'Set the first element of the third added np to be the second-to-last field in the solution:
        Edge(1) = x(NOF - 1):
        'Set the second element of the third added np to be the first field in the solution:
        Edge(2) = x(1)
    End If
Else
    'Set the first element of the third added np to be the field preceeding the moving field in the solution:
    Edge(1) = x(S(1) - 1):
    'Set the second element of the third added np to be the field succeding the moving field in the solution:
    Edge(2) = x(S(1) + 1)
End If
Attribute6 = Edge
End Function

'This is the tabu search function.
Public Function TabuSearch(x, NOF, NOP, SP, MijMatrix, PijMatrix, CTij, Nuij, TimeLimit)
Dim T1(), T2(), T3(), T4(), T5(), T6() As Long 'Define tabu lists.
Dim TL1, TL2, TL3, TL4, TL5, TL6 As Integer 'Define tenures.
TL1 = 3: TL2 = 3: TL3 = 3: TL4 = 3: TL5 = 3: TL6 = 3
If CheckBox6.Value = True Then 'Do if user has checked the "Assign tabu list lengths" checkbox in the UI.
        TL1 = Worksheets("Start").Range("AC32").Value: TL2 = Worksheets("Start").Range("AC33").Value:
        TL3 = Worksheets("Start").Range("AC34").Value
        TL4 = Worksheets("Start").Range("AC35").Value: TL5 = Worksheets("Start").Range("AC36").Value:
```

```
        TL6 = Worksheets("Start").Range("AC37").Value
End If
ReDim T1(1 To TL1, 1 To 2)
ReDim T2(1 To TL2, 1 To 2)
ReDim T3(1 To TL3, 1 To 2)
ReDim T4(1 To TL4, 1 To 2)
ReDim T5(1 To TL5, 1 To 2)
ReDim T6(1 To TL6, 1 To 2)
Dim Aspiration() As Double 'Define aspiration function.
ReDim Aspiration(1 To NOF, 1 To NOF)
For i = 1 To NOF 'Dimension the aspiraton function.
    For j = 1 To NOF
        Aspiration(i, j) = -10 ^ 30
    Next j
Next i
'Define variables for average improvement per iteration between ejection chains:
Dim AverageImpr, GrandAverageImpr As Double
Dim Improvement, BigImprovement As Single
'Define variables for various iteration counters:
Dim Breakaways, NoEjectionChains, NoRandomRestarts, k, NoImpr, NoBigImpr, MaxNoImpr, MaxNoBigImpr, xprev() As Long
'Define a starting solution:
ReDim xprev(1 To NOF)
'Define a matrix containing values from the search to be reported:
Dim ReportMatrix() As Single
ReDim ReportMatrix(1 To 65000, 1 To 6)
BestOV = OV(x, NOF, NOP, PijMatrix, CTij)
VeryBestOV = BestOV
'Set the starting solution to be the input solution (x is the input solution):
xprev = x
AverageImpr = 0.00000000001
BestAverageImpr = 0.00000000001
NumberOfGAIUpdates = 0
NumberOfStarts = 1
'Get the number of nonimproving iterations allowed before an ejection chain (ejc) is applied from the UI.
MaxNoImpr = Worksheets("Start").Range("AC29").Value
'Get the number of nonimproving ejcs allowed before random restart from the UI.
MaxNoBigImpr = Worksheets("Start").Range("AC30").Value
'Do this while the search is within the UI specified time limit.
While TimeSerial(Hour(Now()), Minute(Now()), Second(Now())) < TimeLimit
    k = k + 1
    If AverageImpr < BestAverageImpr Then 'Do if the average improvement per iteration is
    'worse than when the best solution was found.
        If NoBigImpr > MaxNoBigImpr Then 'Do if the number of nonimproving ejcs is too large.
            'Add 1 to the random restart counter:
            NoRandomRestarts = NoRandomRestarts + 1
            'Generate a new random starting solution:
            xprev = RndSubset(NOF)
            'Reset variables used to compute average improvement:
            NumberOfImprovements = 0: AverageImpr = 0
            'Empty the tabu lists:
            Init = InitTabuLists(T1, T2, T3, T4, T5, T6, TL1, TL2, TL3, TL4, TL5, TL6)
            'Reset nonimproving iterations counters and compute objective function
            'value (ov) of the new starting solution:
            NoImpr = 0: NoBigImpr = 0: BestOV = OV(xprev, NOF, NOP, PijMatrix, CTij)
        End If
    End If
    If NoImpr > MaxNoImpr Then 'Do if too many nonimproving iterations have occured:
        NoEjectionChains = NoEjectionChains + 1 'Add 1 to ejc counter.
        xprev = EjectionChain(xprev, NOF, NOP, SP, CTij, Nuij) 'Apply ejc-function to current solution.
        NoImpr = 0 'Reset nonimproving iteration counter.
        BestOV = OV(xprev, NOF, NOP, PijMatrix, CTij) 'Compute the best ov since the application if this ejc.
        NoBigImpr = NoBigImpr + 1 'Add 1 to the number of ejcs without improving the very best ov.
        If BestOV > VeryBestOV Then 'Do if the ov of the solution returned by this ejc is better
        'than the very best ov found during the entire search.
            BigImprovement = BestOV - VeryBestOV 'Compute the improvement to the very best ov.
            verybestx = bestx: VeryBestOV = BestOV 'Set the new very best solution and set the new very best ov.
            NoBigImpr = 0
        End If
    End If
    'Improve solution by applying the shift move function:
    xprev = ShiftNeighbourhood(xprev, NOF, NOP, SP, MijMatrix, PijMatrix, CTij, T1, T2, T3, T4, T5, T6, TL1, TL2,
TL3, TL4, TL5, TL6, Aspiration)
    CurrentOV = OV(xprev, NOF, NOP, PijMatrix, CTij) 'Set the ov of the current solution.
    If CurrentOV > BestOV Then 'Do if the current ov is better than the best ov since last ejc application.
        'If the solution came from a nonimproving solution and has a better ov than found since the
        'last ejc application, it has broken away from a local optimum:
        If NoImpr > 0 Then Breakaways = Breakaways + 1
        NoImpr = 0: Improvement = CurrentOV - BestOV:
        NumberOfImprovements = NumberOfImprovements + 1:
        'Compute an average improvement:
        AverageImpr = (AverageImpr + CurrentOV / Improvement) / NumberOfImprovements
        bestx = xprev: BestOV = CurrentOV 'Reset the best solutionan ov since last ejc application.
        If BestOV > VeryBestOV Then 'Do if ov is better than best overall ov found during the entire tabue search:
            BestAverageImpr = AverageImpr
            BigImprovement = BestOV - VeryBestOV
            verybestx = bestx: VeryBestOV = BestOV
            NoBigImpr = 0
        End If
    Else
        NoImpr = NoImpr + 1 'Add 1 to the noumber of nonimproving iterations since the last ejc application.
    End If


    If CheckBox2.Value = True Then Worksheets("Start").Range("AC21").Value = k 'Post iterations to UI.
    If CheckBox3.Value = True Then 'Post improvements to UI:
        Worksheets("Start").Range("AC23").Value = Improvement: Worksheets("Start").Range("AC24").Value = BigImprovement
    End If
    If CheckBox4.Value = True Then 'Post ov values to UI:
        Worksheets("Start").Range("AC26").Value = CurrentOV:
        Worksheets("Start").Range("AC27").Value = BestOV: Worksheets("Start").Range("AC28").Value = VeryBestOV
    End If

    If CheckBox9.Value = True And k < 65001 Then 'Add values to the report matrix:
        ReportMatrix(k, 1) = k
        ReportMatrix(k, 2) = NoImpr
        ReportMatrix(k, 3) = NoBigImpr
        ReportMatrix(k, 4) = CurrentOV
        ReportMatrix(k, 5) = BestOV
        ReportMatrix(k, 6) = VeryBestOV
    End If
Wend
If CheckBox9.Value = True Then 'Post search history in UI:
    Worksheets("SearchHistory").Range(Worksheets("SearchHistory").Cells(2, 1), Worksheets("SearchHistory").Cells(20001, 6)).ClearContents
    Worksheets("SearchHistory").Range(Worksheets("SearchHistory").Cells(2, 1), Worksheets("SearchHistory").Cells(k + 1, 6)) = ReportMatrix
    Worksheets("SearchHistory").Range("P2").Value = NoRandomRestarts
    Worksheets("SearchHistory").Range("Q2").Value = NoEjectionChains - NoRandomRestarts
    Worksheets("SearchHistory").Range("R2").Value = Breakaways
End If
TabuSearch = verybestx 'Return the best solution found.
End Function

Public Function ShiftNeighbourhood(x, NOF, NOP, SP, MijMatrix, PijMatrix, CTij, T1, T2, T3, T4, T5, T6, TL1, TL2,
TL3, TL4, TL5, TL6, Aspiration)
'A move is defined as a change from one sequence number to another.
'3,4 for example means that the field with sequence number 3 moves to sequence number 4.

Dim S(1 To 2) As Long
Dim NShift()
ReDim NShift(1 To NOF * (NOF - 1), 1 To 3)
Dim Yield, Val, FNo, counter, counter1, counter2, counter3, movecounter As Long
'Genrate all relevant moves and compute objective function values
'associated with these moves:
For counter1 = 1 To NOF
    For counter2 = 1 To NOF
        movecounter = movecounter + 1
        If counter1 = counter2 Then
            counter2 = counter2 + 1
            If counter2 > NOF Then
                Exit For
            End If
        End If
        xstar = x
        If counter1 < counter2 Then
            For counter3 = counter1 To counter2 - 1
                xstar(counter3) = x(counter3 + 1)
            Next counter3
            xstar(counter2) = x(counter1)
        End If
        If counter1 > counter2 Then
            For counter3 = counter2 + 1 To counter1
                xstar(counter3) = x(counter3 - 1)
            Next counter3
            xstar(counter2) = x(counter1)
        End If
        NShift(movecounter, 1) = OV(xstar, NOF, NOP, PijMatrix, CTij)
        NShift(movecounter, 2) = counter1
        NShift(movecounter, 3) = counter2
    Next counter2
Next counter1
BestValue = -10 ^ 30
Dim BestMove(1 To 2)
Dim BestNumber As Long
For counter = 1 To NOF * (NOF - 1)
    'The best possible shift move is selected
    For counter1 = 1 To NOF * (NOF - 1)
        If NShift(counter1, 1) > BestValue Then
            BestValue = NShift(counter1, 1)
            BestNumber = counter1
            BestMove(1) = NShift(counter1, 2)
            BestMove(2) = NShift(counter1, 3)
        End If
    Next counter1

    'Tabu status is checked
    Dim Pass1, Pass2, Pass3, Pass4, Pass5, Pass6 As Integer
    Pass1 = 0: Pass2 = 0: Pass3 = 0: Pass4 = 0: Pass5 = 0: Pass6 = 0:
    e1 = Attribute1(BestMove, x, NOF)
    Tabu = 0
    For counter1 = 1 To TL1
        If e1(1) = T1(counter1, 1) Then
            If e1(2) = T1(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL2
        If e1(1) = T2(counter1, 1) Then
            If e1(2) = T2(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL3
        If e1(1) = T3(counter1, 1) Then
            If e1(2) = T3(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    If Tabu = 0 Then
        Pass1 = 1
    ElseIf BestValue > Aspiration(e1(1), e1(2)) Then
        Pass1 = 1
    End If

    e2 = Attribute2(BestMove, x, NOF)
    Tabu = 0
    For counter1 = 1 To TL1
        If e2(1) = T1(counter1, 1) Then
            If e2(2) = T1(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL2
        If e2(1) = T2(counter1, 1) Then
            If e2(2) = T2(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL3
        If e2(1) = T3(counter1, 1) Then
            If e2(2) = T3(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    If Tabu = 0 Then
        Pass2 = 1
    ElseIf BestValue > Aspiration(e2(1), e2(2)) Then
        Pass2 = 1
    End If

    e3 = Attribute3(BestMove, x, NOF)
    Tabu = 0
    For counter1 = 1 To TL1
        If e3(1) = T1(counter1, 1) Then
            If e3(2) = T1(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL2
        If e3(1) = T2(counter1, 1) Then
            If e3(2) = T2(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL3
        If e3(1) = T3(counter1, 1) Then
            If e3(2) = T3(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    If Tabu = 0 Then
        Pass3 = 1
    ElseIf BestValue > Aspiration(e3(1), e3(2)) Then
        Pass3 = 1
    End If

    e4 = Attribute4(BestMove, x, NOF)
    Tabu = 0
    For counter1 = 1 To TL4
        If e4(1) = T4(counter1, 1) Then
            If e4(2) = T4(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL5
        If e4(1) = T5(counter1, 1) Then
            If e4(2) = T5(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL6
        If e4(1) = T6(counter1, 1) Then
            If e4(2) = T6(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    If Tabu = 0 Then
        Pass4 = 1
    ElseIf BestValue > Aspiration(e4(1), e4(2)) Then
        Pass4 = 1
    End If

    e5 = Attribute5(BestMove, x, NOF)
    Tabu = 0
    For counter1 = 1 To TL4
        If e5(1) = T4(counter1, 1) Then
            If e5(2) = T4(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL5
        If e5(1) = T5(counter1, 1) Then
            If e5(2) = T5(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL6
        If e5(1) = T6(counter1, 1) Then
            If e5(2) = T6(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
```

```
    If Tabu = 0 Then
        Pass5 = 1
    ElseIf BestValue > Aspiration(e5(1), e5(2)) Then
        Pass5 = 1
    End If

    e6 = Attribute6(BestMove, x, NOF)
    Tabu = 0
    For counter1 = 1 To TL4
        If e6(1) = T4(counter1, 1) Then
            If e6(2) = T4(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL5
        If e6(1) = T5(counter1, 1) Then
            If e6(2) = T5(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    For counter1 = 1 To TL6
        If e6(1) = T6(counter1, 1) Then
            If e6(2) = T6(counter1, 2) Then Tabu = 1: Exit For
        End If
    Next counter1
    If Tabu = 0 Then
        Pass6 = 1
    ElseIf BestValue > Aspiration(e6(1), e6(2)) Then
        Pass6 = 1
    End If
    'If all neighbouring pairs have a pass status, the move is accepted
    'and the search moves on with the corresponding solution:
    If Pass1 + Pass2 + Pass3 + Pass4 + Pass5 + Pass6 = 6 Then
        Exit For
    'If not all neighbouring pairs have a pass status the solution is brought
    'out of contention by assigning a large bad objective function value to it:
    Else
        NShift(BestNumber, 1) = -10 ^ 30
        BestValue = -10 ^ 30
    End If
    'If there are no more solutions left in the neighbourhood, the
    'incoming solution to the shift neighbourhood is selected and sent to the tabu search:
    If counter = NOF * (NOF - 1) Then
        ShiftNeighbourhood = x: NoImpr = MaxNoImpr
        Exit Function
    End If
Next counter

'At this stage solution has been selected,
'and it is thus time to update all tabu lists and aspiration values.
COV = OV(x, NOF, NOP, PijMatrix, CTij)
CurAsp = Aspiration(e1(1), e1(2))
Aspiration(e1(1), e1(2)) = Application.WorksheetFunction.Max(CurAsp, COV, BestValue)
For counter1 = 1 To TL1 - 1
    T1(counter1, 1) = T1(counter1 + 1, 1)
    T1(counter1, 2) = T1(counter1 + 1, 2)
Next counter1
T1(TL1, 1) = e4(1)
T1(TL1, 2) = e4(2)

CurAsp = Aspiration(e2(1), e2(2))
Aspiration(e2(1), e2(2)) = Application.WorksheetFunction.Max(CurAsp, COV, BestValue)
For counter1 = 1 To TL2 - 1
    T2(counter1, 1) = T2(counter1 + 1, 1)
    T2(counter1, 2) = T2(counter1 + 1, 2)
Next counter1
T2(TL2, 1) = e5(1)
T2(TL2, 2) = e5(2)

CurAsp = Aspiration(e3(1), e3(2))
Aspiration(e3(1), e3(2)) = Application.WorksheetFunction.Max(CurAsp, COV, BestValue)
For counter1 = 1 To TL3 - 1
    T3(counter1, 1) = T3(counter1 + 1, 1)
    T3(counter1, 2) = T3(counter1 + 1, 2)
Next counter1
T3(TL3, 1) = e6(1)
T3(TL3, 2) = e6(2)

CurAsp = Aspiration(e4(1), e4(2))
Aspiration(e4(1), e4(2)) = Application.WorksheetFunction.Max(CurAsp, COV, BestValue)
For counter1 = 1 To TL4 - 1
    T4(counter1, 1) = T4(counter1 + 1, 1)
    T4(counter1, 2) = T4(counter1 + 1, 2)
Next counter1
T4(TL4, 1) = e1(1)
T4(TL4, 2) = e1(2)

CurAsp = Aspiration(e5(1), e5(2))
Aspiration(e5(1), e5(2)) = Application.WorksheetFunction.Max(CurAsp, COV, BestValue)
For counter1 = 1 To TL5 - 1
    T5(counter1, 1) = T5(counter1 + 1, 1)
    T5(counter1, 2) = T5(counter1 + 1, 2)
Next counter1
T5(TL5, 1) = e2(1)
T5(TL5, 2) = e2(2)

CurAsp = Aspiration(e6(1), e6(2))
```

```
Aspiration(e6(1), e6(2)) = Application.WorksheetFunction.Max(CurAsp, COV, BestValue)
For counter1 = 1 To TL6 - 1
    T6(counter1, 1) = T6(counter1 + 1, 1)
    T6(counter1, 2) = T6(counter1 + 1, 2)
Next counter1
T6(TL6, 1) = e3(1)
T6(TL6, 2) = e3(2)

'Set the new solution to be output:
xstar = x
If BestMove(1) < BestMove(2) Then
    For counter3 = BestMove(1) To BestMove(2) - 1
        xstar(counter3) = x(counter3 + 1)
    Next counter3
    xstar(BestMove(2)) = x(BestMove(1))
End If
If BestMove(1) > BestMove(2) Then
    For counter3 = BestMove(2) + 1 To BestMove(1)
        xstar(counter3) = x(counter3 - 1)
    Next counter3
    xstar(BestMove(2)) = x(BestMove(1))
End If

ShiftNeighbourhood = xstar 'Return the selected solution.

End Function

Public Function EjectionChain(x, NOF, NOP, SP, CTij, Nuij)
Dim EJ(), i, j, k, Used() As Long
Dim Potential As Single
ReDim EJ(1 To NOF, 1 To 2), Used(1 To NOF)
CuttingJD = CT(x, NOF, NOP, SP, CTij)
NuijMin = 10 ^ 30

For i = 1 To NOF
    j = PeriodFunction(CuttingJD(i) - 7 * (SP - 1))
    If Nuij(x(i), j) < NuijMin Then NuijMin = Nuij(x(i), j): EJ(1, 1) = i: EJ(1, 2) = j: Potential = 1 - Nuij(x(i), j) 'The field with the
    'highest potential to improve is selected for ejection.
Next i
Used(EJ(1, 1)) = 1
Potential = 1 - NuijMin
NumberOfEjections = 1
DeltaMax = -10 ^ 30
Success = 0
For counter = 2 To NOF
    For i = 1 To NOF
        If Used(i) = 0 Then
            k = PeriodFunction((CuttingJD(i) - 7 * (SP - 1)))
            'Each field's current profit coefficient is compared to what it would recieve in the ejection period:
            Delta = Nuij(x(i), EJ(counter - 1, 2)) - Nuij(x(i), k)
            If Delta > 0 - Potential Then
                If Delta > DeltaMax Then
                    'If a field is selected, it is recorded with position and period on the ejection list EJ:
                    DeltaMax = Delta
                    'This value stores the position of field x(i).
                    EJ(counter, 1) = i
                    'This value stores the period number of the start of the slot that was opened up by the ejection of field x(i).
                    EJ(counter, 2) = k
                End If
            End If
        End If
    Next i
    'If there is no ejection that moves a field to a better position, the ejection chain ends.
    'A field may draw from the available potential, thus may be allowed to move if its decrease
    'in relative profit is less than the available potential.
    If DeltaMax <= 0 - Potential Then
        Exit For
    Else
        DeltaMax = -10 ^ 30
        Success = Success + 1
        Used(EJ(counter, 1)) = 1
        'Potential = Potential + DeltaMax
    End If
Next counter
FirstEjectedField = x(EJ(1, 1))
For counter = 1 To Success
    x(EJ(counter, 1)) = x(EJ(counter + 1, 1))
Next counter
x(EJ(Success + 1, 1)) = FirstEjectedField
EjectionChain = x
End Function
Public Function InitTabuLists(T1, T2, T3, T4, T5, T6, TL1, TL2, TL3, TL4, TL5, TL6)
'This function initialises the tabu lists.
For j = 1 To 2
    For i = 1 To TL1
        T1(i, j) = 0
    Next i
    For i = 1 To TL2
        T2(i, j) = 0
    Next i
    For i = 1 To TL3
        T3(i, j) = 0
    Next i
    For i = 1 To TL4
        T4(i, j) = 0
    Next i
    For i = 1 To TL5
        T5(i, j) = 0
    Next i
    For i = 1 To TL6
        T6(i, j) = 0
    Next i
Next j
InitTabuLists = 1
End Function

Public Function PeriodFunction(JD) 'This function returns the week number given a Julian date.
    Dim P As Integer
    P = 1 + (JD - 4) / 7
    If P > 52 Then P = 52

    PeriodFunction = P
End Function
```

A.3 Visual Basic for Excel code for command button "Generate plan"

```
Private Sub CommandButton2_Click() 'This sub is only here to generate a nice-looking printout.
'Generate Plan
HF = Worksheets("Start").Range("Z22").Value
NOF = Application.WorksheetFunction.CountIf(Worksheets("FDB").Range("A2:A999"), HF)
TNOF = Application.WorksheetFunction.CountA(Worksheets("Start").Range("B2:B999"))
'This again accounts for harvested fields:
For counter1 = 1 To TNOF
    If Worksheets("FDB").Cells(counter1 + 1, 1).Value = HF Then
        RemainingFraction = 1
        If Worksheets("FDB").Cells(counter1 + 1, 19).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 21).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 22).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 24).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 25).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 27).Value / 100
        End If
        If RemainingFraction <= 0 Then
            NOF = NOF - 1
        End If
    End If
Next counter1
LOS = Worksheets("Start").Range("Z21").Value - Worksheets("Start").Range("Z19").Value
FDS = Worksheets("Start").Range("Z19").Value - Worksheets("Start").Range("Z2").Value
LDS = Worksheets("Start").Range("Z21").Value - Worksheets("Start").Range("Z2").Value
SP = PeriodFunction(FDS)
EP = PeriodFunction(LDS)
NOP = EP - SP + 1
RVP = Worksheets("Start").Range("Z8").Value
counter2 = 0
Dim MijMatrix()
Dim FieldNamesVector() As Variant
Dim CTAij() As Single
ReDim FieldNamesVector(1 To NOF)
ReDim MijMatrix(1 To NOF, 1 To NOP)
ReDim CTAij(1 To NOF)
For counter1 = 1 To TNOF
    Mass = 0
    If Worksheets("FDB").Cells(counter1 + 1, 1).Value = HF Then
        RemainingFraction = 1
        If Worksheets("FDB").Cells(counter1 + 1, 19).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 21).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 22).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 24).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 25).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 27).Value / 100
        End If
        If RemainingFraction > 0 Then
            counter2 = counter2 + 1
            For counter3 = 1 To NOP
                FieldNamesVector(counter2) = Worksheets("FDB").Cells(counter1 + 1, 2).Value
                Area = Worksheets("FDB").Cells(counter1 + 1, 3).Value * RemainingFraction
                Yield = Worksheets("FDB").Cells(counter1 + 1, 85 + SP + counter3).Value
                Mass = Mass + Area * Yield
            Next counter3
            TotArea = TotArea + Area
            TotYield = TotYield + Mass / NOP
        End If
    End If
Next counter1
'Cutting times matrix CTAij and CTYij
counter2 = 0
For counter1 = 1 To TNOF
```

```
    If Worksheets("FDB").Cells(counter1 + 1, 1).Value = HF Then
        RemainingFraction = 1
        If Worksheets("FDB").Cells(counter1 + 1, 19).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 21).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 22).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 24).Value / 100
        End If
        If Worksheets("FDB").Cells(counter1 + 1, 25).Value = "Harvest" Then
            RemainingFraction = RemainingFraction - Worksheets("FDB").Cells(counter1 + 1, 27).Value / 100
        End If
        If RemainingFraction > 0 Then
            counter2 = counter2 + 1
            Area = Worksheets("FDB").Cells(counter1 + 1, 3).Value
            CTAij(counter2) = Application.WorksheetFunction.RoundDown(Area / TotArea * LOS, 3)
        End If
    End If
Next counter1

If CheckBox1.Value = True Then
    CTij = CTAij
Else
    CTij = CTYij
End If
Dim CuttingPeriod As Integer
Dim SortedPlan()
ReDim SortedPlan(1 To NOF)
Dim FieldNames()
ReDim FieldNames(1 To NOF)
Dim xvector()
ReDim xvector(1 To NOF)
For counter = 1 To NOF
    FieldNames(counter) = Worksheets("Sequence").Cells(2, counter).Value
    xvector(counter) = Worksheets("Sequence").Cells(1, counter).Value
Next counter
Worksheets("HarvestPlan").Cells.ClearContents
For counter = 1 To NOF
    SortedPlan(counter) = FieldNames(xvector(counter))
Next counter
Dim Plan()
ReDim Plan(1 To NOF + 1, 1 To 22)
CuttingJD = CT(xvector, NOF, NOP, SP, CTij)
For counter = 1 To NOF
    Field = SortedPlan(counter)
    For counter1 = 1 To TNOF
        If Worksheets("FDB").Cells(counter1 + 1, 1).Value = HF Then
            If Worksheets("FDB").Cells(counter1 + 1, 2).Value = Field Then
                FieldRow = counter1 + 1:
                Exit For
            End If
        End If
    Next counter1
    Area = Worksheets("FDB").Cells(FieldRow, 3).Value
    Accs = Worksheets("FDB").Cells(FieldRow, 6).Value
    Vrty = Worksheets("FDB").Cells(FieldRow, 7).Value
    AgeBrn = (CuttingJD(counter) - Worksheets("FDB").Cells(FieldRow, 9).Value + Worksheets("Start").Range("Z2").Value) / 30.4375
    CuttingPeriod = PeriodFunction(CuttingJD(counter))
    TnHa = Worksheets("FDB").Cells(FieldRow, 86 + CuttingPeriod).Value
    Tons = TnHa * Area
    RV = Worksheets("FDB").Cells(FieldRow, 30 + CuttingPeriod).Value
    RRV = Worksheets("FDB").Cells(FieldRow, 248 + CuttingPeriod).Value
    MRV = Worksheets("FDB").Cells(FieldRow, 194 + CuttingPeriod).Value
    Cost = Worksheets("FDB").Cells(FieldRow, 140 + CuttingPeriod).Value
    EV1 = Worksheets("FDB").Cells(FieldRow, 19).Value
    Ed1 = Worksheets("FDB").Cells(FieldRow, 20).Value
    Ep1 = Worksheets("FDB").Cells(FieldRow, 21).Value
    EV2 = Worksheets("FDB").Cells(FieldRow, 22).Value
    Ed2 = Worksheets("FDB").Cells(FieldRow, 23).Value
    Ep2 = Worksheets("FDB").Cells(FieldRow, 24).Value
    EV3 = Worksheets("FDB").Cells(FieldRow, 25).Value
    Ed3 = Worksheets("FDB").Cells(FieldRow, 26).Value
    Ep3 = Worksheets("FDB").Cells(FieldRow, 27).Value
    Plan(counter + 1, 1) = Field: Plan(counter + 1, 2) = Area: Plan(counter + 1, 3) = Accs
    Plan(counter + 1, 4) = Vrty: Plan(counter + 1, 5) = MonthName(Month(CuttingJD(counter) + Worksheets("Start").Range("Z2").Value), True)
    Plan(counter + 1, 6) = AgeBrn: Plan(counter + 1, 7) = TnHa
    Plan(counter + 1, 8) = Tons: Plan(counter + 1, 9) = RV: Plan(counter + 1, 10) = RRV
    Plan(counter + 1, 11) = MRV: Plan(counter + 1, 12) = Cost: Plan(counter + 1, 13) = EV1
    Plan(counter + 1, 14) = Ed1: Plan(counter + 1, 15) = Ep1: Plan(counter + 1, 16) = EV2
    Plan(counter + 1, 17) = Ed2: Plan(counter + 1, 18) = Ep2: Plan(counter + 1, 19) = EV3
    Plan(counter + 1, 20) = Ed3: Plan(counter + 1, 21) = Ep3: Plan(counter + 1, 22) = CuttingJD(counter)
Next counter
Plan(1, 1) = "Field": Plan(1, 2) = "Area": Plan(1, 3) = "Accs": Plan(1, 4) = "Vrty": Plan(1, 5) = "Month":
Plan(1, 6) = "AgeBrn": Plan(1, 7) = "TnHa"
Plan(1, 8) = "Tons": Plan(1, 9) = "RV": Plan(1, 10) = "RRV": Plan(1, 11) = "MRV": Plan(1, 12) = "Cost": Plan(1, 13) = "Ev1"
Plan(1, 14) = "Ed1": Plan(1, 15) = "Ep1": Plan(1, 16) = "Ev2": Plan(1, 17) = "Ed2": Plan(1, 18) = "Ep2": Plan(1, 19) = "Ev3"
Plan(1, 20) = "Ed3": Plan(1, 21) = "Ep3": Plan(1, 22) = "CutJD"
Worksheets("HarvestPlan").Range(Worksheets("HarvestPlan").Cells(1, 1), Worksheets("HarvestPlan").Cells(NOF + 1, 22)).Value = Plan
Worksheets("HarvestPlan").PrintPreview (False)
End Sub

Public Function PeriodFunction(JD) 'This function returns the week number given a Julian date.
    Dim P As Integer
    P = 1 + (JD - 4) / 7
    If P > 52 Then P = 52
    PeriodFunction = P
End Function
```

APPENDIX B

The accompanying compact disc

The *compact disc* (CD) accompanying this dissertation contains the dissertation in pdf format as well as the computer implementation of the DSS presented in this dissertation in the macro-activated Excel workbooks format "*.xlsm". There are a number of folders on the CD and the contents of these folders are described here by their folder names.

Dissertation. This folder contains the dissertation in pdf format.

DSS Implementation. This folder contains the latest version of the DSS implementation, in Excel format. The file is unprotected and may be edited completely. The file and its VBA-code may be activated by double-clicking the file icon, subsequently clicking the "Options" button in the "**Security Warning**" area within the Excel window, selecting "Enable this content" and finally clicking the "OK" button. This file may now be used as described in §8.2. The only known problem occurs when the solver is run in such a way as to continue past midnight; the time limit does not work properly in that case.

20-Field Experiment. This folder contains the Excel files used in the verification experiment in Table 8.10 and should not be used for any other purposes than to test results presented in §8.4. The VBA code within these files has been password-protected to prevent unforeseen problems, and it is recommended that macros are NOT ENABLED for these files. In order to perform re-runs of the experiments the macros must, however, be activated at one's own risk.

40-Field Experiment. This folder contains the Excel files used in the verification experiment in Table 8.11 and should not be used for any other purposes than to test results presented in §8.4. The VBA code within these files has been password-protected to prevent unforeseen problems, and it is recommended that macros are NOT ENABLED for these files. In order to perform re-runs of the experiments the macros must, however, be activated at one's own risk.

2009 Hindsight. This folder contains the Excel files used in the hindsight scheduling of 2009 shown in Table 9.6 and further discussed in §9.2.1 and should not be used for any other purposes than to test results presented in §9.2.1. The VBA code within these files has been password-protected to prevent unforeseen problems, and it is recommended that macros are NOT ENABLED for these files. In order to perform re-runs of the experiments the macros must, however, be activated at one's own risk.