KTH Computer Science
and Communication

Mobile traffic dataset comparisons through cluster analysis of radio network event sequences

Att jämföra trafikdatamängder för mobila enheter genom klusteranalys för sekvenser av event i radionätet

BJÖRN LÖFROTH
(BLOFROTH@KTH.SE)

Master's Thesis in Computer Science at CSC
Supervisor: Giampiero Salvi
Examiner: Anders Lansner
Collaborating company: Ericsson
2014-03-01

TRITA xxx yyyy-nn

Abstract

Ericsson regularly collects traffic datasets from different radio networks around the world. These datasets can be used for several research purposes, ranging from general statistics to more specific studies such as system troubleshooting and buffer-level analysis. Currently, a researcher may find it difficult to assess if a certain dataset is useful for a particular investigation, since there exists no easily accessible overview of the properties of the different datasets.

This thesis project aims to make it easier to compare the existing traffic datasets in terms of general statistics, user and time coverage, data integrity and the patterns of sequences in radio network event logs. The key contribution is a method of clustering event sequences based on sequence duration and occurrences of a number of key events. A method called the *Gap-statistic* was applied to determine that using 11 clusters was suitable for the analysis, although no strong evidence was found for the existence of well separated clusters.

The results show that the method can work as a useful extension of basic comparative statistics. Two dense ranges of sequence durations discovered in the basic statistics could successfully be linked to corresponding clusters of sequences. Extensive statistics about the cluster members then revealed detailed properties of the sequences in these two dense areas, at a deeper level than could be understood from the basic statistics.

A problematic part of interpreting the results of the method is that many different perspectives of the data need to be considered at the same time to find interesting links. Future work could include automating the process of linking features in the basic statistics to clusters.

Referat

Att jämföra trafikdatamängder för mobila enheter genom klusteranalys för sekvenser av event i radionätet

Ericsson samlar regelbundet in trafikdatamängder ifrån olika radionätverk runt om i världen. Dessa datamängder kan användas i många olika forsknings- och utvecklingssyften, både ur ett generellt perspektiv genom att betrakta allmän statistik, men även för specifika studier som till exempel felsökning av system och analys av buffernivåer i nätverket. För närvarande kan det dock vara svårt för en potentiell analytiker av dessa datamängder att avgöra om de lämpar sig för en viss studie.

Detta examensarbete är inriktat på att underlätta jämförelser mellan olika inspelningar av dessa trafikdatamängder vad gäller allmän statistik, användar- och tidstäckning och dataintegritet samt mönster i loggarna för radionätshändelser. Det huvudsakliga bidraget av detta examensarbete är en metod för att klustra händelsesekvenser baserat på deras tidsspann och antal förekomster av nyckelhändelser. Den s.k. *Gap Statistic*-metoden användes för att avgöra att 11 kluster var lämpligt för klusteranalysen, även om starka bevis inte kunde hittas för existensen av tydligt separerade kluster i de studerade datamängderna.

Resultaten visar på att den valda metoden kan fungera som en användbar fördjupning av allmän jämförande statistik. Två intervall av tätt samlande durationer för händelsesekvenser kunde länkas till två motsvarande kluster av sekvenser. Utförlig statistik om sekvenserna i dessa kluster kunde visa på sekvensernas egenskaper i stor detalj, på en djupare nivå än vad som kunde åstadkommas med allmän statistik.

En problematisk del i tolkandet av metodens resultat var att flera olika perspektiv av data var tvungna att betraktas på samma gång för att kunna upptäcka intressanta länkar. En vidareutveckling av arbetet i denna rapport kan vara att skapa metoder för att automatisera och förenkla processen att länka intressanta fenomen i den allmänna statistiken till olika kluster.

Contents

Acknowledgements

This report describes the work of a degree project for the School of Computer Science and Communication (CSC) at the Royal Institute of Technology (KTH) in Stockholm, Sweden. The project has been undertaken in collaboration with Ericsson AB, that provided an initial project idea and kindly provided access to the data used in this work, as well as equipment and office space for me during the project.

I would like to thank my academic supervisor, Associate Professor Giampiero Salvi at the department of Speech, Music and Hearing in the CSC institution at KTH, for the valuable feedback I have received on both the taken approach and the report drafts throughout the project.

At Ericsson, I would like to thank my two domain expert supervisors, Tomas Lundborg and Eric Andersson at the Development Unit Radio Systems & Technology group in Kista, for their valuable insight into the radio network domain, giving me feedback about the project and also helping me with practical issues regarding the datasets and computer environments at Ericsson. The feedback I have received from the rest of the group at Ericsson, as well as other Ericsson attendees to my company presentations, is also much appreciated.

Björn Löfroth, Stockholm, Sweden
February 27, 2014

List of acronyms

CDF Cumulative Distribution Function

CN Core Network

CS Circuit Switched

CSV Comma Separated Values

DPI Deep Packet Inspection

GSM Global System for Mobile Communications

HSPA High-Speed Packet Access

IMEI International Mobile Equipment Identity

IMSI International Mobile Subscriber Identity

IP Internet Protocol

ISDN Integrated Services Digital Network

M2M Machine to Machine

OS Operating System

PDP Packet Data Protocol

PS Packet Switched

PSTN Public Switched Telephone Network

RAN Radio Access Network

RANAP RAN Application Part

RBS Radio Base Station

RNC Radio Network Controller

RRC Radio Resource Control

TAC Type Allocation Code

UE User Equipment

UMTS Universal Mobile Telecommunications System

URA UTRAN Registration Area

Chapter 1

Introduction

Ericsson regularly conducts measurements in different mobile networks and collects log data about what kind of data traffic is sent and what communication protocol events are registered in the network as devices connect, communicate and disconnect from the network. The resulting datasets can be used for several research purposes, ranging from general statistics about usage of different services and traffic volumes, to more specific studies such as system troubleshooting and buffer-level analysis.

One problem that exists for current and potential future analysts of these datasets is that it is difficult to assess if a particular dataset is useful for a certain investigation. There can be big differences in the data characteristics between different recording episodes, owing to the fact that the collection may have taken place in different radio networks, in different years and for longer or shorter periods. There might also be limitations in the collected data, such as missing data for some periods. To address this, an overview of the qualities and limitations of each dataset, and how they differ from each other is needed.

One perspective on the collected datasets is to examine the sequences of events that are registered in the network as a user device connection proceeds. It is not well known to what extent the patterns of such sequences differ between different radio networks. Factors such as the network configuration and what devices and services are popular could be likely to cause different sequence patterns to be observed in different networks.

1.1 Aim

This master thesis project aims to propose methods for the automatic collection of aggregated information from existing datasets, with the goal of generating a good overview of their properties. The aim can be divided into three parts:

A. Describing dataset metadata: user and time coverage, and data integrity

B. Describing basic statistics

C. Investigating patterns of common radio network event sequences through clustering, which in turn can be subdivided into two parts:

  - Determine if there exists any global sequence patterns that are common across all datasets
  - Evaluate the usefulness of event sequence clustering results in comparing datasets properties, in relation to using basic statistics

The purpose of the two descriptive parts (A) and (B) is to provide a good overview of what the studied datasets contain, what basic features they have and how their features compare and differ. By looking at this part, a potential analyst should e.g. be able to quickly discard the dataset if the time covered is too short, or if there is not enough variance in a specific metric between datasets to motivate a certain study.

The analysis of common sequences (C) aims to complement the first two parts by providing a more in-depth view on how the datasets are similar and different, from the perspective of radio network event sequences.

1.2 Reader guidance

A background part with information relating to the structure and function of radio networks is presented in Chapter 2, while the theoretical background needed to understand the statistical methods used in this thesis is presented in Chapter 3. The theory part includes descriptions of some relevant methods when analyzing large datasets statistically and a presentation of the body of theory relating to clustering, which is used for the analysis of common sequences.

In Chapter 4, the used method is described in detail. The used datasets are described in the chapter, and the concept of a radio network event sequence is explained and defined. The three main parts of the chapter describe the detailed methods of how the three aims A, B, and C are pursued, respectively.

The results achieved when applying the described methods is presented in Chapter 5. This includes the key results from the basic dataset analysis and several different perspectives on the features of the groupings found during the analysis of common sequences.

The results and the method is discussed in Chapter 6, while conclusions and ideas for future work are given in Chapter 7.

Chapter 2

Background

2.1 Radio networks

The radio networks considered in this report are Universal Mobile Telecommunications System (UMTS) networks, so called 3G networks. UMTS is the successor to the Global System for Mobile Communications (GSM) standard, and provides higher bit-rates and more flexibility in supporting multiple applications such as voice and video calls, multimedia streaming and online games. In addition, the considered networks also support High-Speed Packet Access (HSPA) which gives a further boost in transmission data rates in both the uplink and downlink.

2.1.1 UMTS network architecture

The basic idea of cellular radio networks such as UMTS is that the covered area is divided into cells. Stationary transceiver nodes, Radio Base Stations (RBSs), are placed to cover some number of cells each. Another important component of the network is the Radio Network Controller (RNC) which handles relaying of information to a Core Network. Each RNC is responsible for a number of RBSs. The RNCs and RBSs collectively form the Radio Access Network (RAN).

An overview of the structure of the RAN can be seen in Figure 2.1. The core UMTS components and terminology are explained below [1]:

- **User Equipment (UE)**: The UE is the device connecting to the radio network. It could be e.g. a cell phone, tablet, router or possibly some specialized device like a credit-card reader.

- **Node B**: The name Node B is used for the radio base stations in UMTS. These can communicate over the air with the UE:s.

- **Radio Network Controller (RNC)**: Each RNC is responsible for a number of Node B:s – its domain. The RNC owns and controls the radio resources in its domain and is the service access point for the services the RAN provides to the Core Network.

- **Core Network**: The Core Network is responsible for switching and routing calls and data connections to external networks. It has two separate domains: Packet Switched (PS) and Circuit Switched (CS). The PS domain handles Internet Protocol (IP) based traffic, while the CS domain handles communication where a static route (circuit) needs to be set up, e.g. for voice calls. Examples of external circuit switched networks include the phone network, Public Switched Telephone Network (PSTN), and Integrated Services Digital Network (ISDN) which combines voice and data transmissions.

The communication between the RAN and the Core Network is done over the RAN Application Part (RANAP) protocol, which handles things such as paging a user (e.g. for a voice call), tracking the UE location, and performing hard hand overs, i.e. when the communication from a UE needs to go through another RNC, perhaps since the UE has moved geographically.

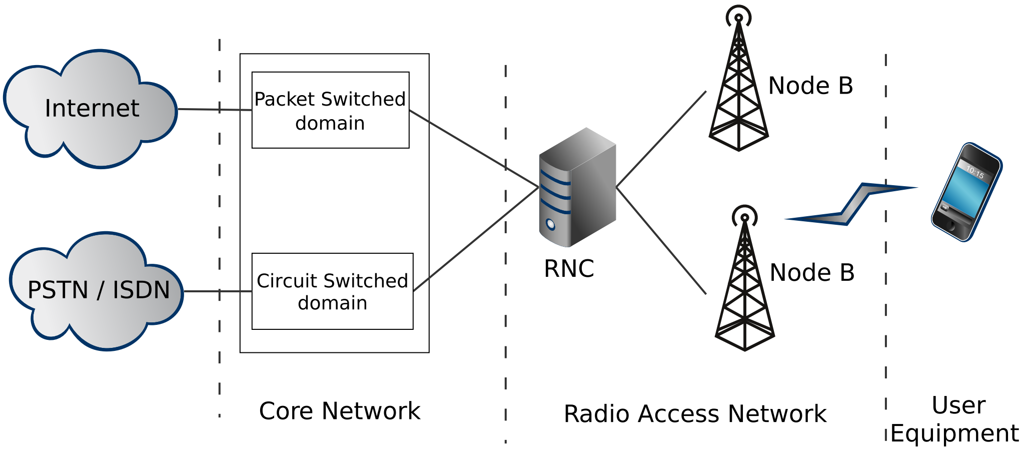

Figure 2.1: UMTS RAN network structure

The UE and RAN communicate over the Radio Resource Control (RRC) protocol. This protocol handles, among other things, the relaying of information and requests toward the Core Network, the setup and release of the RRC connection between the UE and RAN, transmission of signal measurement reports and communication about changes in the radio channel configuration for the UE.

2.1.2 Communication states

A UE can either be in *idle* or *connected* mode. In idle mode, the UE chooses a suitable cell and monitors its control channel. To move to connected mode, the

Table 2.1: RRC service state characteristics

	User data bit-rate		
State	Uplink	Downlink	Power consumption (mA) †
Cell DCH	High	High	200-300
Cell FACH	Low	Low	100-150
Cell PCH	- ‡	- ‡	< 5
URA PCH	- ‡	- ‡	< 5

† The power consumption numbers are taken from [2]
‡ No user data communication possible

UE establishes an RRC connection. The connected mode is further divided into four RRC service states: *Cell DCH*, *Cell FACH*, *Cell PCH*[1] and *URA PCH*. The state names are derived from at which geographical granularity the UE is known at (cell/URA) in the state, which will be explained in more detail further on, and which underlying downlink transport channel is used (DCH, FACH, PCH), which we omit the details about here, but is well described in e.g. [1].

The possible transitions between these modes and states are presented in Figure 2.2. These states have different characteristics e.g. in terms of how much power they consume from the UE, the attainable data rates and how the location of the UE is tracked. An overview over their differences is presented in Table 2.1.

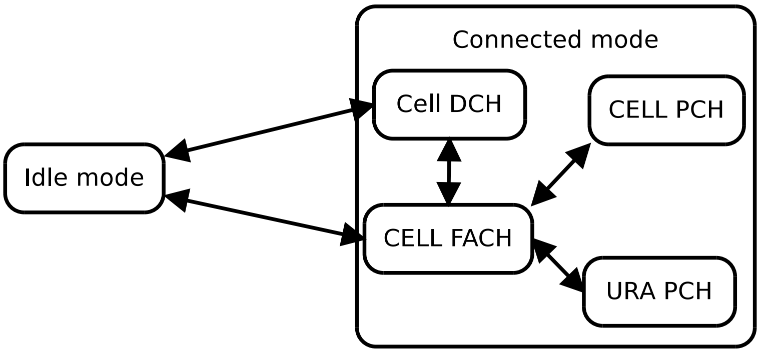

Figure 2.2: RRC service state transitions

2.1.3 Mobility

As a UE moves between cells, the RAN and the Core Network need to keep track of its location, at least at such a degree that they know to which RNCs to direct a

[1] The *Cell PCH* is defined by 3GPP, but in an Ericsson RAN the state is omitted, since the *URA PCH* state achieves similar functionality.

paging request (e.g. for a voice call). The mechanisms for tracking the location of a UE vary depending on which state it is in [1]:

- **Cell DCH**: In this state a dedicated radio channel is maintained to the UE. The UE can be communicating with up to four cells at the same time. The currently communicating cells for a UE are called its *active set*. Using signal strength reports from the UE, the RNC decides when cells are to be added or removed from the active set. This is communicated to the UE by the *soft handover* procedure.

- **Cell FACH**: The UE is in this state only communicating with one cell. Whenever a UE finds a more suitable cell, the *RRC cell update* procedure is run which lets the RAN know where it can be reached.

- **Cell PCH**: The UE is in this state only communicating with one cell, and only monitors downlink channels, unable to use the uplink. Instead, whenever a more suitable cell is found, the UE first has to change state to Cell FACH, and then run the *RRC cell update* procedure, after which it can switch back to Cell PCH.

- **URA PCH**: This state is similar to Cell PCH, but the difference is that here an update is only done when the UTRAN Registration Area (URA) changes, and the procedure is called *RRC URA update*. URAs are used to group cells into larger groups. The effect is that the update is done more rarely, requiring less resources from the UE and RAN.

Furthermore, the faster HSPA downlink transport channel, HS-DSCH, can be used in the Cell DCH state if the UE supports it and it is within range of a cell that supports it [3]. In this case, the UE is only communicating with one cell on the downlink, the so-called serving HS-DSCH cell. The procedure to change the serving HS-DSCH cell involves the RNC, which means it is also aware of the current serving HS-DSCH cell of a UE.

2.1.4 User and device identification

Each UE can be identified with two main numbers: the International Mobile Subscriber Identity (IMSI) and International Mobile Equipment Identity (IMEI). The IMSI can be thought of as the phone number of a subscription. In all log types used in this report an anonymized IMSI is used, which cannot be tied to a real phone number.

The IMEI identifies the actual physical device and can be thought of as a combined model and serial number for a mobile device. The first 8 digits constitute the Type Allocation Code (TAC) code [4], which only can be linked to the device model, and not the unique part for each device. In the log types analyzed in this report, only the TAC part of the IMEI is available, if at all.

2.2 Log types

The log types considered in this report can be divided into two groups: radio network event logs and IP-traffic logs. The former cover logging related to RAN protocol events processed by the RNC, while the latter follow IP-traffic as collected at a point in the Core Network.

2.2.1 Radio network event logs

As a UE connects to the RAN, various protocols interact to setup, modify and tear down communication between the different nodes in the network. The RNC has a central role in this process, and it may log different events relating to a user connection. The logged events cover both RANAP and RRC protocol events, and also other events relating to the connection such as when the communication is switched between a higher bit rate and a slower bit rate channel, or when the user moves between cells. In this report this log type will simply be referred to as *event* logs.

The events are logged in a binary format, where each entry has a timestamp at millisecond resolution, event type, some parameters for tracking which sequence of events belongs to the same user, and some parameters that describe the event in more detail.

With the help of an Ericsson developed tool the binary logs can be translated into a text based Comma Separated Values (CSV) format. The tool tries to link each event to its IMSI, whenever possible, and can also output extra information such as during which intervals a user was in a specific channel.

2.2.2 IP traffic logs

An Ericsson developed Deep Packet Inspection (DPI) tool can be connected at different points in the Core Network to analyze packet switched data during the measurement period. It records important characteristics at varying levels of detail, and classifies the traffic type. In this project, four logs produced by this tool will be analyzed: flow logs, summary logs, packet header logs, and to a lesser extent the Packet Data Protocol (PDP) logs.

Flow logs An IP flow is a sequence of packets where the so called 5-tuple (source IP, source port, destination IP, destination port, protocol type) is the same and there is no gap longer than 60 seconds in between any two consecutive packets. In the case of a TCP flow, it is initiated with a three-way handshake and ended either by a FIN packet or the 60-second timeout. In addition to the identifying 5-tuple, the logs also contain start and end time at millisecond resolution, an anonymized IMSI, the device part of the IMEI, and traffic tags provided by the DPI tool.

Summary logs In this log type, IP statistics are aggregated per minute into summary activities, with one entry for each application that was recognized by the DPI tool for a user during that minute.

Packet header logs This log is similar to the flow logs, but instead has one entry per packet sent or received by the UE. The log tracks the time the packet was received at the logging node, and also whether the packet was sent on the uplink or downlink.

Packet Data Protocol (PDP) logs This log can be used to match IP addresses with IMSI numbers. It is only used in this work to map each IMSI to an IMEI number, which allows analysis of device types.

Chapter 3

Theory

This chapter contains a brief description of the theory relevant for this project. The algorithms used to deal with statistical analysis of large data quantities are described in Section 3.1, while the background theory for clustering methods is presented in Section 3.2.

3.1 Statistical algorithms for large data quantities

3.1.1 Welford's algorithm for mean and variance

Care has to be taken when calculating mean and variance on large datasets to maintain floating point precision. The straightforward way to calculate mean, μ, and variance, σ^2, of a set of values x_1,...,x_N is:

$$\begin{aligned} \mu &= (\sum_{i=1}^{N} x_i)/N \\ \sigma^2 &= \frac{\sum_{i=1}^{N} x_i^2 - (\sum_{i=1}^{N} x_i)^2/N}{N} \end{aligned}$$

Implementing this with standard floating point arithmetic can however lead to loss of precision and other serious problems, such as negative variance [5]. A method without these flaws is described by Knuth [5] (who cites Welford [6]) using the recurrence formulas:

$$\begin{aligned} M_1 &= x_1 \\ M_k &= M_{k-1} + (x_k - M_{k-1})/k \\ S_1 &= 0 \\ S_k &= S_{k-1} + (x_k - M_{k-1}) \times (x_k - M_k) \end{aligned}$$

with $\mu = M_N$ and $\sigma^2 = S_n/(n-1)$.

3.1.2 Reservoir sampling

The problem of extracting a fixed number, K, of random samples from a stream of items, with N items in total, is not trivial to solve effectively if N is not known beforehand. One solution is to simply save all items and then pick every $\frac{N}{K}$ sample, but this consumes a substantial amount of memory, especially if each item takes up much space in memory. Knuth [5] describes an algorithm dubbed *Reservoir sampling* that solves this problem.

The original algorithm is stated in terms of processing a file of unknown size, but it is easy to convert it to the case of processing items from a stream in memory, which is described in Algorithm 1. We here assume `randint(i,j)` generates a random integer between i and j (inclusive) and that the stream `s` has two methods: `hasNext()` which returns `True` if there are more items in the stream and `False` otherwise; and `next()` which returns the next item in the stream.

The general procedure of the algorithm is to first fill up the sample reservoir with the first K items, and with probability K/n include each new item, thus replacing an existing item.

It can be proven by induction that, in each iteration of the loop, all previously seen items have an equal probability of being sampled to the reservoir [7].

Algorithm 1: Reservoir sampling

```
Data: Number of samples $K$, stream s
Result: Array of samples R
R ← Array();
for $i \leftarrow 1$ to $K$ do
    if s.hasNext() then
        R[$i$] ← s.next() ;
    else
        break;
    end
end
$n \leftarrow 0$;
while s.hasNext() do
    $n \leftarrow n + 1$;
    $i_r$ ← randint($1, n$);
    if $i_r \leq K$ then
        R[$i_r$] ← s.next() ;
    end
end
```

3.2 Clustering

The field of clustering is a very typical representative of what is called *unsupervised* machine learning. As opposed to *supervised* machine learning, which tries to learn models from known correct pairs of input and output data, in unsupervised learning, no such mapping is known. Instead, an unsupervised method tries to find hidden structure in unlabelled data.

In clustering, the hidden structure we are looking for is a grouping that puts similar elements together and dissimilar elements in different groups. In this thesis the focus will be on hard clustering, i.e. when each element belongs to only one group (cluster).

3.2.1 K-means clustering

One of the most popular clustering algorithms is usually just known as K-means, but it is perhaps more appropriate to talk about the K-means problem. The algorithms that solve the problem are slightly different but are generally characterised by their simplicity and speed.

The K-means problem is, given N data points $\vec{x}_1, ..., \vec{x}_N$, each of dimensionality M, to split them into K sets (clusters) S_1,...,S_K to minimize the within cluster sum of squares:

$$\sum_{i=1}^{k} \sum_{\vec{x}_j \in S_i} ||\vec{x}_j - \vec{\mu}_i||^2 \tag{3.1}$$

where μ_i is the mean of points in S_i.

It has been shown that the K-means problem is NP-hard [8], i.e. there cannot exist an algorithm that solves it optimally in polynomial running time in K, N and M asymptotically [1]. The popular algorithms for solving the K-means problem therefore only look for solutions that are approximately correct, while still managing to have polynomial running time.

An algorithm that has been commonly used for the K-means problem was first described by Lloyd [9]. It is often referred to as Lloyd K-means. Pseudo-code for this algorithm is presented in Algorithm 2 (adapted from a restatement in [10]). The distance metric is typically chosen to be the squared Euclidean distance between two points in $\mathbb{R}^M$:

$$d(\vec{x}, \vec{y}) = \sum_{i=1}^{M} (x_i - y_i)^2 \tag{3.2}$$

The algorithm first initalizes the cluster centers to some positions (described in the next section) and then in iterative cycles assigns points to the cluster with the

[1] Unless P=NP, which has not been proven. However, it is generally assumed, within the field of Computer Science, that the equality does not hold.

nearest center and moves each cluster center to the mean of all of its members.

Algorithm 2: Lloyd K-Means

```
Data: Number of clusters $K$, data points $X = \{\vec{x}_1, ..., \vec{x}_N\}$
Result: Clusters, i.e. sets of points, $S_1$,...,$S_K$ such that each $x_i$ is a member
        of exactly one $S_j$
changed ← True ;
$C$ ← InitClusterCenters();
while changed = True do
    foreach $\vec{c}_j \in C$ do
        $S_j \leftarrow \{\vec{x}_i \in X | j = \underset{k}{\operatorname{argmin}}\, d(\vec{x}_i, \vec{c}_k)\}$ ;
    end
    foreach $\vec{c}_j \in C$ do
        $\vec{c}_j \leftarrow \frac{1}{|S_j|} \sum_{\vec{x}_i \in S_j} \vec{x}_i$ ;
    end
    if the clusters have changed in this iteration then
        changed = True ;
    else
        changed = False ;
    end
end
```

3.2.2 Cluster center initialization

Commonly, the initial cluster centers (provided by the `InitClusterCenters()`-function in the pseudo-code of Algorithm 2) would be chosen at random uniformly from the given data points in X. The method is however only guaranteed to find a local minimum for Equation 3.1, which means that the result depends heavily on the initialization. There has been some work on providing an approximation bound for Lloyd K-means type algorithms, mostly by initializing the cluster centers in clever ways. One such method, called K-means++, uses random adaptive sampling to provide a solution that has been proved to be a $O(\log k)$ approximation of the optimum in expectation [10]. The authors also present empirical data that suggests that K-means++ initialization both speeds up the running time and improves the solution quality for Lloyd K-means clustering on some real and synthetic datasets.

The algorithm for K-means++ cluster center initialization is presented in Algorithm 3. Here $D(\vec{x}) = \underset{k}{\operatorname{argmin}}\, d(\vec{x}, \vec{c}_k)$ where k goes up from 1 to the last cluster index chosen.

```
Algorithm 3: K-Means++ cluster center initialization
  Data: Number of clusters $K$, data points $X = \{\vec{x}_1, ..., \vec{x}_N\}$
  Result: Initial positions of the clusters centers $C = \vec{c}_1, ..., \vec{c}_K$
  Choose $\vec{c}_1$ at random uniformly from $X$ ;
  for $k = 2$ to $K$ do
      Choose $c_k$ at random from $X$, where each $x_i$ has probability $\frac{D(\vec{x_i})^2}{\sum_{\vec{x_j} \in X} D(\vec{x_j})^2}$
      of being chosen ;
  end
```

Since the initialization is still non-deterministic we can get different results when running the algorithm multiple times. An easy way to get a better solution according to Equation 3.1 is simply to run the combination of K-means++ initialization and Lloyd K-means several times and pick the solution with the minimum error.

3.2.3 Determining K

One of the early methods of determining the number of clusters to use, K, is the so called *elbow method.* As stated in [11]:

> Start with k=1, and keep increasing it, measuring the cost of the optimal quality solution. If at some point the cost of the solution drops dramatically, that's the true k.

Cost here refers to the error of a clustering. The dramatic drop will on a plot make the curve have the shape of an elbow, hence the name. The method is quite simple, but also ambiguous, since "drops dramatically" is very vague.

A more rigorous method, the *gap statistic* [12], captures the intuition of the elbow method, while giving a specific recommendation for which K to use. The general idea is to compare the result of the quality of a clustering on the given data to the average quality when clustering on some data generated to represent unclusterable data, sampled from a chosen *reference distribution*, and choose the k for which the gap in quality is the largest. The authors state that it is meant to be used when there are well separated clusters in the data.

To describe how the gap statistic measures the quality of a clustering, we use the same notation as before, i.e. we have clusters $S_1, ..., S_K$ such that each data point x_i is a member of exactly one S_c. We also define the sum of pairwise distances within cluster c as [12]:

$$D_c = \sum_{i,j \in S_c} d(\vec{x}_i, \vec{x}_j)$$

and also a dispersion measure for a complete clustering with k clusters:

$$W_k = \sum_{c=1}^{k} \frac{1}{2|S_c|} D_c$$

When the squared Euclidean distance (Equation 3.2) is used, W_k becomes the pooled within-cluster sum of squares around the cluster centers, and can then be calculated as in Equation 3.1. The authors then define:

$$Gap_n(k) = E_n^*\{\log(W_k)\} - \log(W_k)$$

Here E_n^* denotes the expectation over samples of size n from the reference distribution. The authors use the gap statistic with two reference distributions: (a) uniform over the range of the observed values, and (b) uniform over a box aligned with the principal components of the data. The empirical results in [12] suggest that (b) gives the best results overall. The algorithm for sampling from a type (b) reference distribution is presented in Algorithm 4.

Algorithm 4: Sampling from a reference distribution aligned with the principal components of the data

```
Data: Data matrix $X$, size $n \times m$, with points as rows. Number of points to
      sample, $s$.
Result: Sample point matrix, $S$, size $s \times m$
1. Find the singular value decomposition: $X = UDV$
2. Project points onto eigenvectors: $X' = XV^T$
3. Find the minimum and maximum values in each of the columns (rotated
dimensions) in $X'$
4. Build $S'$, with the values in each column being sampled uniformly between
the minimum and maximum found for each respective column in the
previous step.
5. Compute the real coordinates of the sampled points as $S = S'V$
```

The expectation $E_n^*\{\log(W_k)\}$ is computed as the average $\log(W_k)$ when the clustering algorithm is run on B different samples from the reference distribution, using k clusters. With σ_k being the standard deviation for $\log(W_k)$ in these B runs, the authors also define the term:

$$s_k = \sqrt{1 + 1/B}\sigma_k \tag{3.3}$$

Putting all of this together to determine the best k according to the gap statistic, we have the following steps:

1. Cluster the given data X for each considered number of clusters $k = 1, 2, ..., K$, recording each within-cluster dispersion measure W_k.

2. Generate B datasets, each by sampling from the reference distribution (e.g. as in Algorithm 4).

3. For each considered k, cluster the B datasets, and record the within-cluster dispersion for each sample and k pair as W_{kb}^*.

   We then have:

$$Gap(k) = (1/B)\sum_{b=1}^{B} \log(W_{kb}^*) - \log(W_k)$$

4. The best k according to the gap statistic is then, using s_k from Equation 3.3:

$$\hat{k} = \text{the smallest } k \text{ such that } Gap(k) \geq Gap(k+1) - s_{k+1}$$

3.2.4 Feature normalization

The scale of the features selected as dimensions for the input data to K-means greatly affects the outcome of the clustering. If, e.g., feature A is measured in seconds and feature B in milliseconds, this will have the effect that we consider the same absolute time difference more important for feature B than for feature A. In the case where the features are measuring different aspects altogether, such as time and distance, the relative scale of the features is more difficult to reason about intuitively. Even if raw features are used, this is an indirect decision about the relative importance of the features.

A solution can then be to normalize (sometimes called standardize) the features, so that e.g. they all have the same range, usually between 0 and 1. This is not without critique however. If each feature is allowed to contribute equally to the distance measure, this can remove implicit importance judgments, but it can also remove meaningful differences [13].

One simple method of normalizing a feature is to linearly scale its values to the $[0, 1]$-range [14]. Using

$$x_{min} = \min_i x_i$$
$$x_{max} = \max_i x_i$$

the transformation

$$\bar{x} = \mathbf{lintf}(x) = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3.4}$$

then puts $\bar{x}$ in the range [0,1].

Chapter 4

Method

The work in this thesis can be divided into three logical parts, corresponding to the three aims listed in Section 1.1: extracting dataset metadata, extracting basic statistics and analyzing common radio network event sequences. The goal has been to implement a reusable set of tools that can produce HTML reports on the similarities and differences between datasets from these three perspectives.

Seen from a chronological point of view, the method has three phases: (1) data extraction (2) data analysis (3) HTML report generation. An overview of the kind of data that is processed and from which the data reports are generated is presented in Figure 4.1.

The studied datasets are described in Section 4.1. The term *radio network event sequence* is defined and motivated in Section 4.2. The details of the method relating to aims A, B and C are presented in Section 4.3, 4.4 and 4.5, respectively.

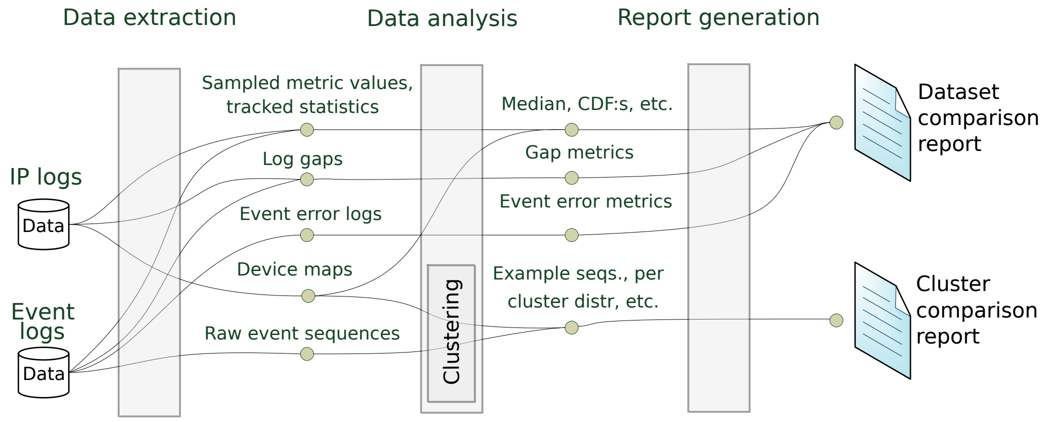

Figure 4.1: An overview of the data extracted and how it is used in the two different reports.

Table 4.1: Dataset attributes

Label	Region	Year	# RNCs
ASIA-A	Asia	2010	4 †
NA	North America	2012	1
ASIA-B	Asia	2013	2
EU	Europe	2012	2 †

† Only 10% of users considered

4.1 Datasets

In this report, datasets from four collection episodes are considered. Time constraints prohibited analysis of more datasets. The selected datasets were chosen to have a wide geographical spread while spanning a few years. In some cases the size of the dataset made it infeasible to consider all users for the basic statistic extraction. In these cases only a certain fraction of all users were considered. See Section 4.4.3 for a short motivation that we still maintain statistical properties when considering fewer users. The datasets are further described in Table 4.1, which specifies for which datasets a smaller fraction of the users was considered. The term *dataset* will be used to refer to the data from one of the collection episodes listed in the table.

4.2 Definition of radio network event sequence

Both the event sequence analysis and the basic statistics rely on the notion of a *radio network event sequence*, which is applicable to only the event logs. It is meant to capture the process of a UE going from *idle* to *connected* mode (see Section 2.1.2), communicating though the RAN and then going back to *idle* mode. Some timeout limitations had to be introduced to limit the memory consumption and increase the performance of the extraction script. For this report, a radio network event sequence is defined to have the following properties:

- The sequence starts with an RRC Connection Request event and contains only one such event
- The events in the sequence are logged on the same RNC
- The events in the sequence are from the same IMSI
- If the sequence contains an IU Release [1] event, events after the the first [2] IU

[1] This marks the termination of the link on the RNC-CN interface (IU), for a user.

[2] In some cases there would be several IU releases recorded in a sequence that satisfies the other conditions, which motivated the need to be restrictive after the first seen IU release

release are only included until there is a gap 180 seconds or more between events

- It is the longest possible sequence under the above conditions that starts with a given RRC Connection Request.

Sequences are then extracted in chronological order from the events in the log file. For performance reasons, the timeouts are not checked after each new event is processed, but instead regularly every 180th second.

A test was run on 5 hours of the Asia dataset without these timeouts to see how many sequences would be cut short. Assuming sequence time-outs are discovered immediately, only 1.5% of sequences were affected in the test, which was deemed acceptable. Since timeouts are only checked every 180th second, we can expect a slightly lower fraction of sequences with missed events.

4.3 Extracting dataset metadata

The time span of each log type is for this report defined to be the time difference between the first and last record of the log. In addition to this, a basic data integrity check was done for all log types by monitoring the gaps present in the logs. This was achieved by recording all time intervals where no log records appeared, if the interval was longer than *dt* seconds. For the *Event*, *Flow* and *Packet* log types, a 15 second *dt* value was picked, to mark an interval not to short to be an acceptable difference between two records during low traffic, while still not being too large which could lead to that legitimate gaps would be missed. For the *Summary* log type, records are made every minute, and a 61 second *dt* value was therefore picked, which will indicate any missing minutes as gaps.

In the case of the radio network event logs, the analysis of gaps can be subdivided over different entities that record events. One level is to consider which RNCs are producing events in the log at any given time. Further subdivided, each RNC has different processing units that split the recording of events among them. As is discussed further on, during high load, logging of events can be turned off for such a processor, which is why it can be interesting to study when there are gaps for a specific processor.

The found gaps in the event logs will be analyzed in terms of the *activity level* over time, which is simply the fraction of processors that have no gap in the log at a point in time. We will consider a) activity levels of the log as a whole, i.e. a fraction out of all processors on all RNCs, and b) activity level per RNC.

The coverage over users in the different log types in each dataset was studied by considering sizes and overlaps between the observed sets of users. To get an idea about the user coverage within a dataset, the user sets from all log types were compared, and subsequently the size of the overlaps was counted. In the case where only 10% of the users were considered (see Section 4.1), the set of all users could still be efficiently extracted, and therefore also be part of this analysis.

Event log data integrity analysis

The event logging mechanism in the RNC is designed to automatically scale down and possibly turn of the logging if the load on a particular processor of the RNC becomes too high. There can also be other factors that prohibit logging and the saving of log files. To verify that the data integrity of the event logs in a particular dataset is good it is therefore necessary to analyze these error intervals. Fortunately, an event is always logged when the logging level is changed for a processor on an RNC. The error intervals per processing unit on each RNC was tracked for later analysis.

4.4 Basic dataset statistics

4.4.1 Extracting distributions

To be able to model the distribution of different metrics in the datasets, some functionality for processing statistics was implemented. Each value of a metric that was selected to be measured was processed in the following way:

1. Mean and variance is tracked by stepwise calculation of the recurrence formulas for Welford's algorithm, as described in Section 3.1.1.

2. The value is considered for inclusion in a sample reservoir, as described in Section 3.1.2. The number of samples was limited to $N = 500000$ to achieve a good balance of statistical accuracy and limiting storage space.

3. After collection of all values is finished, order statistics (i.e. median and different percentiles) were calculated on the sample reservoir.

4. A Cumulative Distribution Function (CDF) is then calculated based on the sample reservoir.

Simple category counts were also tracked, but for this there is no need for sampling since counts take up little storage space.

4.4.2 Selected metrics

After studying the literature related to the radio network theory in Section 2.1 and discussing with some Ericsson researchers, a list of metrics to measure was selected. Because of the relative simplicity of measuring many metrics on the same data, the list is quite long and not all of the metrics will be analyzed in this report. The metrics that will be discussed in this report are presented below, while the rest can be found in Appendix C. Each selected metric is described below, with a short motivation of why it is interesting. The metrics marked with † are extracted as category counts, while the others are extracted as distributions as described earlier in this section. Traffic volumes are measured as the total of uplink and downlink volume [3].

The selected metrics by log type:

- Event logs
  - **Total number of events in a sequence**: Indicates how much radio network protocol activity was generated.
  - **Duration of a sequence**: The duration of a sequence could be affected by what type of activity the UE is engaged in, what kind of device it is and user behavior for that activity.
  - **Number of events of four key types in a sequence**: Counts key events per radio network event sequence. This metric is recorded separately for each key event listed below. These key event types were chosen to both cover changes to the communication state and mobility updates (see Section 2.1):
    - Channel switch
    - Soft handover
    - Cell update
    - HS-DSCH cell change
- Summary logs
  - **Summary activity traffic volume**: Measures the number of bytes communicated during a summary activity. This can indicate different usage patterns that might be caused by user behavior or the kind of application being used.
  - **Traffic volume per user**: The total amount of bytes sent or received by a user during the measurement period. This indicates user behavior.
  - **Device types by traffic volume (†)**: A category count of the total amount of bytes sent or received added up for different device types. Device type is one of *HANDHELD*, *M2M*, *PC*, *ROUTER* or *TABLET*. This metric shows which device types are responsible for most of the traffic.
  - **Device OS by traffic volume (†)**: A category count of the total amount of bytes sent or received added up for different device OS's, e.g. *Android*, *iOS*, *Symbian* or *Blackberry*. This metric shows which device OS's are responsible for most of the traffic.
  - **Device type by number of user devices (†)**: A category count of the number of user devices seen in the dataset with a particular device type. This metric shows the device type penetration.

[3] Separate metrics for uplink and downlink traffic are included in Appendix C

  - **Device OS by number of user devices (†)**: A category count of the number of user devices seen in the dataset with a particular OS. This metric shows the device OS penetration.

- Flow logs
  - **Flow duration**: The duration of a flow could be affected by what type of activity the UE is engaged in, what kind of device it is and user behavior for that activity.
  - **Flow data volume**: The number of bytes sent or received during a flow. This metric indicates usage patterns, which could depend on both user and application behavior.

- Packet header logs
  - **Packet size**: The size in bytes of a packet. It can be related to usage patterns in different applications.

4.4.3 Using a fraction of the users

As mentioned in Section 4.1, for two datasets only 10% of the available users were considered in the event logs. A investigation was done to see if the main characteristics were still captured well when fewer users were considered. Six key statistics were collected as described in the previous section for 100%, 10% and 1% of the users in the *ASIA-B* dataset during five hours of log time. The statistics collected were: *Total number of events in a sequence*, *Duration of a sequence* and *Number of events of key types in a sequence*, for each of the four key events described in the previous section.

When the CDFs of these statistics were plotted with the value on a log scale [4] the curves for all three cases were similar enough that one curve covered the others almost completely. This was taken as evidence that using 10% of users still is enough to reliably capture the properties of the dataset. These plots are not included in this report, since the curves were overlapping to such a large extent. Instead, tables over the values at the percentiles 10%, 20%, ..., 90%, for each measured statistic, are included in Appendix A.

4.5 Analysis of common event sequences

The approach chosen to analyze the common radio network event sequences has been to use clustering methods to group them, and then analyze the result both in terms of the identifying features of each cluster, and how the sequences of the considered datasets are divided between these clusters.

The method consists of the following major steps:

[4] The same kind of plot as is used in Section 5.2

Table 4.2: Extracted raw radio network event sequences

Dataset	% of users[†]	# raw sequences
ASIA-A	1 %	7,472,734
NA	10 %	8,110,589
ASIA-B	1 %	14,415,144
EU	10 %	1,290,583

[†] Percentage of the total amount of users appearing in the log, irrespective of whether only 10% was used for basic extraction, as is listed in Table 4.1

1. Extract raw radio network event sequences from each dataset
2. Extract features from the radio network event sequences
3. Transform and normalize features
4. Determine how many clusters, K, to use in the analysis
5. Run the clustering algorithm
6. Analyze the features of the found clusters
7. Generate an HTML report showing the results

The raw data extraction details are outlined in Section 4.5.1. To achieve a reasonable similarity metric a few key features were selected, and then transformed with normalization methods, which is described in Section 4.5.2, corresponding to step 2 and 3 above. The method of applying a cluster algorithm to this data is then given in Section 4.5.3, including how K should be determined.

4.5.1 Raw sequence data extraction and selection

Radio network event sequences from all four considered datasets are used as input to the clustering procedure. In this way, if there are similar sequences across multiple datasets they should end up in the same cluster.

All seen radio network event sequences from a fraction of the users in each dataset were saved for later processing by the clustering method. The storage size for the sequences and the processing time of later steps prohibited analyzing sequences from all the users. Here, only the event timestamps and types were recorded, together with the IMSI number of the device that generated the sequence. The fraction of considered users and the total number of sequences saved per dataset is listed in Table 4.2.

For the clustering, care was taken to select an equal amount of sequences from each dataset so as to not introduce a bias for any dataset. The reservoir sampling

Table 4.3: Selected features and their short names as used in this report

Feature number	Short name	Description
1	duration	Duration of the sequence
2	#chsw	Number of channel switching events in the sequence
3	#soho	Number of soft handover events in the sequence
4	#update	Number of cell update events in the sequence
5	#hschange	Number of HS-DSCH cell change events in the sequence

method (see Section 3.1.2) was used to select $N = 10^6$ of the saved raw sequences from each dataset, for a total of 4×10^6 sequences. Using more sequences proved to take prohibitively long time for running the feature extraction, clustering and cluster analysis.

4.5.2 Feature selection and normalization

The input to the clustering algorithm was constructed by extracting a set of features from each selected raw radio network event sequence in the previous section, and then transforming each feature and normalizing it to be in the range 0-1. The selection of features is important because it defines what we care about for the similarity measure. The transforms then define how we care about these selected features.

Selected features

One can imagine many different things to measure about a sequence, but for this project only a few features were selected to keep the analysis reasonably simple and facilitate interpretation of the resulting clusters. The duration of a sequence (in seconds) was selected as a feature, since it could be used to distinguish between both different user activities and user behavior during these activities.

Furthermore the counts of four key event types were selected as the remaining features: channel switches, soft handovers, cell updates and HS-DSCH cell changes. Channel switches are important, because they indicate if the activity the user is doing has varying communication speed requirements. The last three events indicate user mobility (see Section 2.1.2), in different communication states (see Section 2.1.3), which speaks to different user behaviors.

A list of the selected features and the short names used to refer to them in plots etc is presented in Table 4.3. Using these features means that the data point corresponding to a specific sequence will be a vector with five numeric components, each corresponding to what the extracted feature yields on the specific sequence.

Feature transformation and normalization

As was described in Section 3.2.4, if no normalization is done, the range of each feature greatly affects the distance measure. For this project the features are normalized to the [0,1]-range, to avoid any indirect weighting, which could be hard to reason about. The squared Euclidean distance (Equation 3.2) will be used in the actual clustering, which we will discuss also in this section, to reason about the definition of the distance measure.

Simply scaling each feature linearly to the [0,1]-range (see the **lintf** transform in Section 3.2.4), however, yields very skewed distributions. Histograms of the features after this normalization are presented in Figure 4.2. Note that the Y-axis is in logarithmic scale. We can see that vast majority of sequences have feature values less than 0.1. When initial clustering experiments were conducted with this transform for $K = 10$ clusters, most sequences ended up in one or possibly two clusters, while the remaining clusters only had a few sequences each, that had one or more extreme feature values.

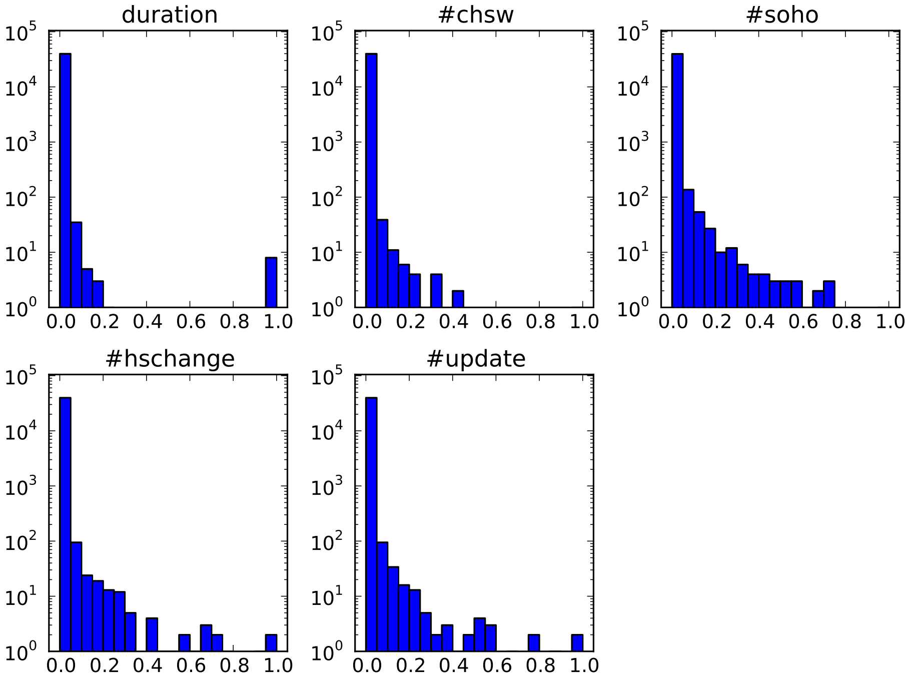

Figure 4.2: Histograms of the selected features after linear scaling to the [0,1]-range. The Y-axis is in logarithmic scale.

To get a better understanding of the distance measure when this transform is used we can consider an example where two points differ only in the duration feature. If two sequences, s_1 and s_2 differ in duration only, the distance between them is simply:

$$d = (u'_1 - u'_2)^2$$

where u'_1 and u'_2 are the linearly scaled durations. Since the scaling is linear, however, the distance between two sequences of duration 5 and 6 seconds will be the same as the distance between two sequences of duration 200 and 201 seconds. It seems appropriate to instead consider the one second as more important in the first case, compared to the second case, since the difference is larger compared to the duration of the sequences. This motivates us to use a feature transform that results in differences scaled to the size of the feature value. For this we can use the natural logarithm, since by multiplying a feature value x with a constant factor k (e.g. $k = 1.1$ to increase the value by 10%) we get a difference only depending on k:

$$\log(kx) - \log(x) = \log(\frac{kx}{x}) = \log(k)$$

This intuitively reasonable property seems to also be appropriate to the count features (feature 2-5 in Table 4.3), where a difference of one event is more important for sequences with fewer such events. Using the log transform by itself, however, does not yield values in the [0,1]-range, but we can apply linear scaling after the log transform. We need to take special care when we have the feature value $x = 0$ since $\log 0 = -\infty$. The method selected to deal with this is to choose an offset, Δd, to mark the value difference between zero and non-zero values, and perform the log transform on the non-zero values, and then linearly scale them in the range $[\Delta d, 1]$. The transform then becomes:

$$x' = \mathbf{logtf}(x) = \begin{cases} 0 & \text{, if } x = 0 \\ \Delta d + (1 - \Delta d) \times \mathbf{lintf}(\log(x)) & \text{, otherwise} \end{cases} \tag{4.1}$$

For the duration feature no offset was used, i.e. $\Delta d = 0$, since it is a continuous feature. For the counts however (feature 2-5), it is significant in itself if an event type appears at all in a sequence. The offset was chosen to be $\Delta d = 0.1$, which gives a somewhat significant difference between zero and non-zero values, while still allowing 0.9 for variances among the non-zero values. When the linear transform is applied the offset corresponds to a difference in feature value that depends on the ratio of the largest and smallest seen feature value. We can see this by expanding the transform for $x > 0$:

$$x'' = \mathbf{lintf}(\log(x)) \tag{4.2}$$

$$= \{\text{Eq. } 3.4\} = \frac{\log(x) - \log(x_{min})}{\log(x_{max}) - \log(x_{min})} \tag{4.3}$$

$$= \frac{\log(\frac{x}{x_{min}})}{\log(\frac{x_{max}}{x_{min}})} \tag{4.4}$$

Now, assuming that $y = kx$ we consider the difference $y'' - x''$:

$$\begin{aligned}
y'' - x'' &= \frac{\log(\frac{kx}{x_{min}})}{\log(\frac{x_{max}}{x_{min}})} - \frac{\log(\frac{x}{x_{min}})}{\log(\frac{x_{max}}{x_{min}})} && (4.5)\\
&= \frac{\log(k)}{\log(\frac{x_{max}}{x_{min}})} + \cancel{\frac{\log(\frac{x}{x_{min}})}{\log(\frac{x_{max}}{x_{min}})}} - \cancel{\frac{\log(\frac{x}{x_{min}})}{\log(\frac{x_{max}}{x_{min}})}} && (4.6)\\
&= \frac{\log(k)}{\log(\frac{x_{max}}{x_{min}})} && (4.7)
\end{aligned}$$

To see the dependency on the range of feature values we can consider the case when the contribution to the feature value is equal between the offset and the transformed and scaled value, in Equation 4.1:

$$\begin{aligned}
x'' \times (1 - \Delta d) &= \Delta d && (4.8)\\
x'' = \frac{\Delta d}{1 - \Delta d} &= \frac{0.1}{0.9} = \frac{1}{9} && (4.9)
\end{aligned}$$

We can note that when $x = x_{min}$ in Equation 4.4 we have $x'' = 0$ since $\log(1) = 0$ in the numerator. Using $x = x_{min}$ and $y = kx$ for some k, we have, by Equation 4.7:

$$y'' - x'' = y'' - 0 = y'' = \frac{\log(k)}{\log(\frac{x_{max}}{x_{min}})} \tag{4.10}$$

We can then solve for k when we have equal contribution from the offset and the value, i.e. when $y'' = \frac{1}{9}$ (from Equation 4.8):

$$\begin{aligned}
y'' = \frac{1}{9} &= \frac{\log(k)}{\log(\frac{x_{max}}{x_{min}})} && (4.11)\\
\Leftrightarrow \frac{1}{9}\log(\frac{x_{max}}{x_{min}}) &= \log(k) && (4.12)\\
\Leftrightarrow (\frac{x_{max}}{x_{min}})^{\frac{1}{9}} &= k && (4.13)
\end{aligned}$$

We now see what the value increase factor is that corresponds to the offset between zero and non-zero values. Some examples for different feature value ranges is presented in Table 4.4.

In summary, using the **logtf** transform assigns at minimum a somewhat significant distance between sequences that contain and do not contain a specific event type, while an increase by a factor k of a feature value within a feature always yields the same contribution to the transformed feature value.

Using the described log transform yields the feature distributions in the histograms presented in Figure 4.3. We can see that the feature histograms are less skewed toward low values as was the case when just the linear transform was used. There is also linearly decreasing trend for the count features, most clearly for *#chsw* and *#soho*. Since a logarithmic scale is used for the Y-axis, this indicates an exponentially decreasing number of sequences for higher feature values.

Table 4.4: Feature value increase corresponding to the non-zero offset value, for different feature value ranges

$\frac{x_{max}}{x_{min}}$	k	% increase
10	1.292	29.2 %
1000	2.154	115.4%
100000	3.594	259.4%

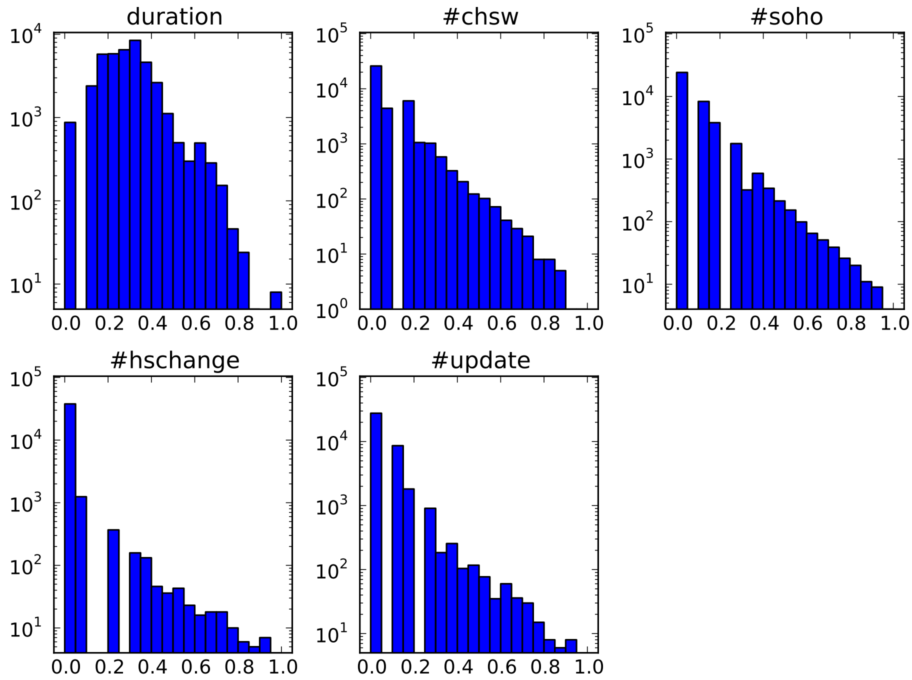

Figure 4.3: Histograms of the selected features after log transform with offsets. The Y-axis is in logarithmic scale.

4.5.3 Clustering

The K-means clustering algorithm (see Section 3.2.1) was selected since it has been well studied and has good performance even for quite large datasets. The K-means++ cluster initialization procedure (see Section 3.2.2) was used in the initialization phase of the algorithm, because it has been shown to improve quality of the solution and reduce running time, while also providing a solution that in expectation is at most $O(\log k)$ from the optimum. The clustering algorithm was repeated 10 times, and the clustering with the best solution according to Equation 3.1 was selected. As was discussed in Section 3.2.2, running the algorithm repeatedly and selecting the best clustering is a simple way of improving the results.

The ELKI clustering software v0.5 [15] was used to do each clustering. The transformed features from the previous section were used as input, and the following software parameters were used:

- `algorithm=clustering.kmeans.KMeansLloyd`
- `kmeans.initialization= KMeansPlusPlusInitialMeans`
- `kmeans.k=`K

To determine which number of clusters K is reasonable to use during the clustering above, the gap statistic method was applied (see Section 3.2.3). The values $K = 1, 2, ..., 20$ were considered. For each possible assignment of K, the clustering (with best-of-10 selection) was run on the transformed features, and then $B = 30$ times on uniformly sampled data as described in the Theory chapter.

Chapter 5

Results

The results achieved using the methods described in the previous chapter are presented below in sections corresponding to the three aims from Section 1.1: dataset metadata (Section 5.1), basic dataset statistics (Section 5.2) and the analysis of common event sequences (Section 5.3).

5.1 Dataset metadata

5.1.1 Time span and log gap analysis

Each of the studied log types in all of the considered datasets was analyzed in terms of its time span and whether there were any gaps in the logs, as described in Section 4.3.

Radio network event logs

An overview over the time span and gap time in the recordings from the different datasets is presented in Table 5.1. The time span is simply the time difference between the first and last timestamp seen in the log. Gap time refers to time where no event was observed from any processing unit on any RNC, added up for periods of at least 15 seconds of duration.

In the table, we can see that the time span of the recordings in the different datasets are quite similar, ranging from just below 8 days up to almost 9 days. All datasets have a total gap time of 2-2.5 hours, or around 1.1% of the time span, except *EU* which has zero gaps.

The activity levels (see Section 4.3) over time in each dataset are presented in Figure 5.1 - 5.3. In these plots, the top part shows the activity level overall in the recording, and for the datasets with multiple RNCs, the other parts show the activity level of individual RNCs. For *Asia-A* (Figure 5.1) we can see that there are two long gaps for the first RNC: first for 14 hours, and in the end of the recording for 24 hours. We also see that all log activity stops 10 times, where 8 of those seem to form a pattern of gaps in 24 hour cycles, occurring regularly every day at around 16:00. Deeper inspection of the data showed that

Table 5.1: Span and gap overview: Event logs

Dataset	Log start	Log end	Time span (days)	Gap time (h)	Gap time %
Asia-A	2010-09-13 15:59	2010-09-22 15:45	8.990	2.50	1.2%
EU	2012-06-25 09:59	2012-07-03 10:59	8.042	0.00	0.0%
NAmerica	2012-03-23 00:59	2012-03-30 17:59	7.708	2.00	1.1%
Asia-B	2013-03-14 06:30	2013-03-22 04:30	7.917	2.00	1.1%

these recurring gaps were all around 15 minutes long. These gaps then constitute the majority of the total 2.5 h of gaps seen in the overview table.

For the *North America* (Figure 5.2) and *Asia-B* (Figure 5.3) datasets we see a similar recurring overall gap in the logs every 24 hours, occurring at around 04:00 and 00:00 respectively. Deeper inspection once again showed that such recurring gaps were around 15 minutes long.

We can also see some other cyclic behaviors: in the *North America* dataset the activity level of one RNC varies from around 1 down to 0 during each day, while we can see that for *Asia-B*, the activity level of the only RNC drops to around 0.9 at similar times of each day. This indicates that the fraction of active processors on some RNCs varies throughout each day. Such recurring patterns could be related to the load of a particular RNC, perhaps if it turns off some processing units when the load is low.

The plot for the *EU* dataset is omitted since it simply shows an activity level at 1.0 with only a handful very small temporary decreases. It was the only dataset of the studied ones without the recurring 15 minute gaps every day.

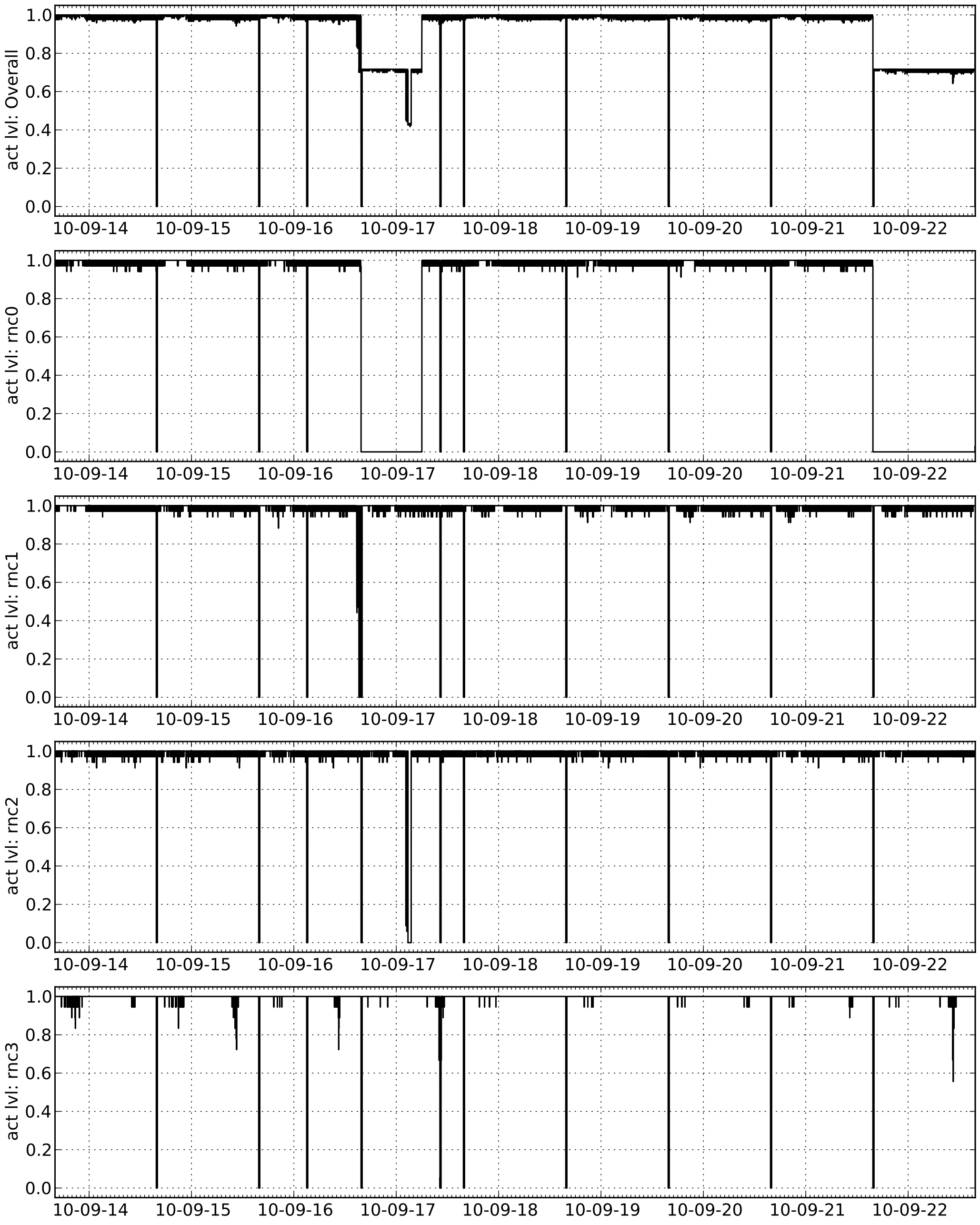

Figure 5.1: Activity level over time: event logs (Asia-A)

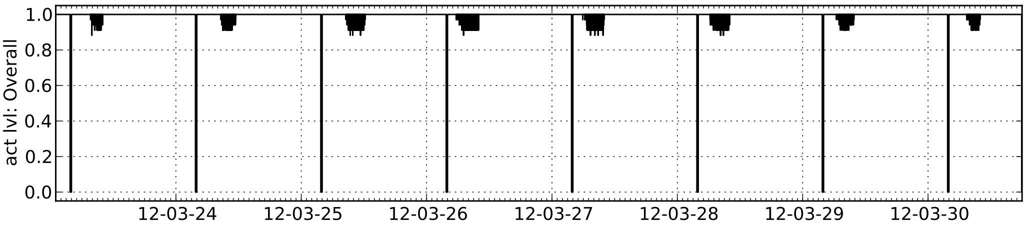

Figure 5.2: Activity level over time: event logs (North America)

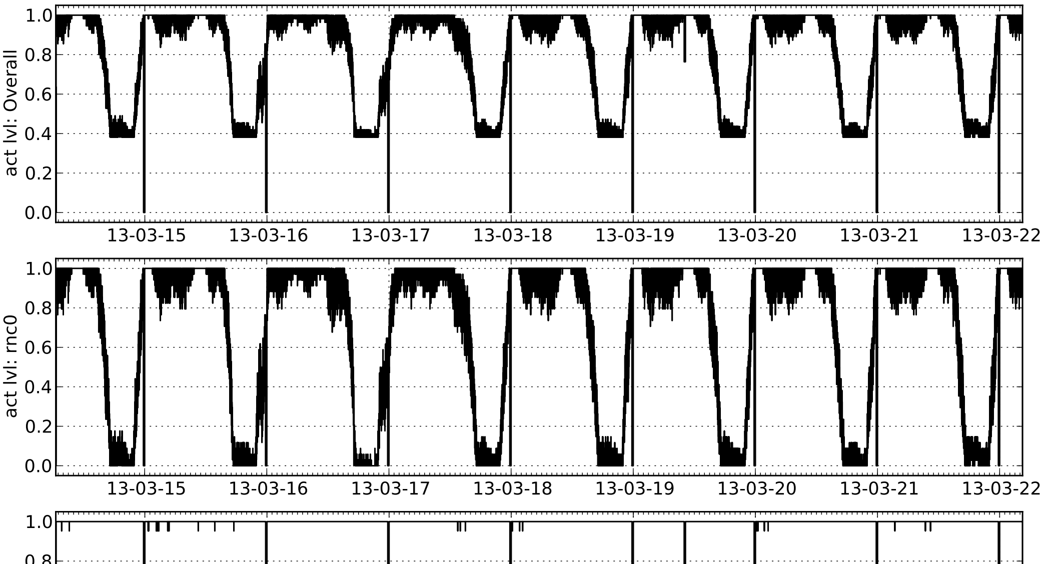

Figure 5.3: Activity level over time: event logs (Asia-B)

Table 5.2: Span and gap overview: Summary logs

Dataset	Log start	Log end	Time span (days)	Gap time (h)	Gap time %
Asia-A	2010-09-16 09:21	2010-09-24 01:08	7.657	0.00	0.0%
EU	2012-06-25 11:37	2012-07-03 12:18	8.028	0.00	0.0%
NAmerica	2012-03-22 15:16	2012-03-30 16:10	8.037	0.00	0.0%
Asia-B	2013-03-18 13:50	2013-03-25 16:10	7.097	0.00	0.0%

Table 5.3: Span and gap overview: Flow logs

Dataset	Log start	Log end	Time span (days)	Gap time (h)	Gap time %
Asia-A	2010-09-14 04:17	2010-09-24 01:08	9.869	8.99	3.8%
EU	2012-06-25 11:37	2012-07-03 11:46	8.006	0.00	0.0%
NAmerica	2012-03-22 15:14	2012-03-30 16:09	8.038	0.00	0.0%
Asia-B	2013-03-18 13:55	2013-03-25 16:04	7.090	0.02	0.0%

Summary logs

An overview table of the time span and gaps in the summary logs, in the different datasets, is presented in Table 5.2. We can see that the time span varies from just over 7 days to just over 8 days. No gaps were observed in these logs.

Flow logs

An overview table of the time span and gaps in the flow logs, in the different datasets, is presented in Table 5.3. Here the time span varies from just above 7 days, for *Asia-B*, to almost 10 days for *Asia-A*. However, *Asia-A* is also the only dataset with a significant amount of gaps, at almost 9 hours (3.8%). The activity level over time for *Asia-A* is presented in Figure 5.4. We can see that a single gap constitutes all of the total gap time, and it starts at 00:00 at 2010-09-16.

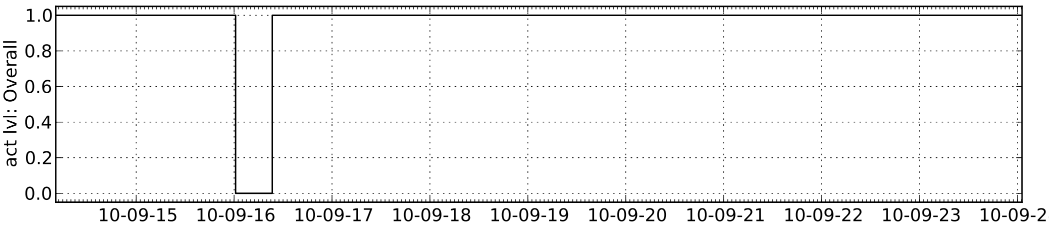

Figure 5.4: Activity level over time: flow logs (Asia-B)

Table 5.4: Span and gap overview: Packet logs

Dataset	Log start	Log end	Time span (h)	Gap time (h)	Gap time %
EU	2012-06-28 17:00	2012-06-28 22:02	5.04	2.96	58.9%
NAmerica	2012-03-28 16:59	2012-03-29 02:03	9.07	6.97	76.8%
Asia-B	2013-03-21 11:00	2013-03-21 12:03	1.05	0.00	0.0%

Packet logs

An overview table of the time span and gaps in the packet logs, in the different datasets[1], is presented in Table 5.4. We can see that for *EU* and *North America* most of the time span is constituted by gaps. The activity levels over time for the *EU* and *North America* datasets are presented in Figure 5.6 and Figure 5.5, respectively. Here we can see that in both datasets, the total gap is accounted for by a long gap in the middle of the time period. In both of these datasets, there is one hour of activity both before and after the gap.

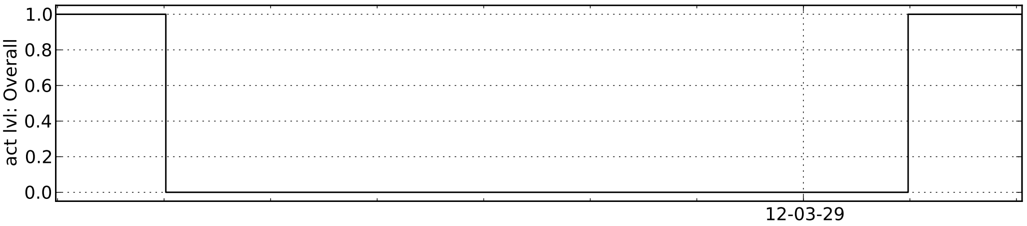

Figure 5.5: Activity level over time: packet logs (North America)

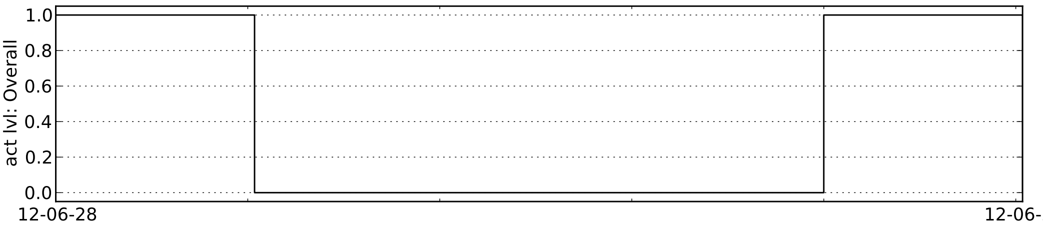

Figure 5.6: Activity level over time: packet logs (EU)

[1] Packet data for the Asia-A dataset is not available

5.1.2 User overlap

In each recording, different sets of users are observed in the various log types. A user is identified by its anonymized IMSI. By comparing which IMSIs appear in which log types in a recording, we can see the user set overlap between them. These overlaps are listed per dataset in Table 5.5-5.8, in decreasing order. The *Overlap size* column refers to the number of users appearing in all the log types listed on that row. The *% of total* column indicates how large part an overlap is out of the total number of unique users observed in all log types for that dataset.

Looking at the tables we can see that there is a variation between the datasets about which log types overlap the most. In *Asia-A* the overlap is largest between flow logs and PDP logs; for *EU* it is between flow and summary logs; while for *North America* and *Asia-B* it is the intersection of the user sets from event and PDP logs. The largest overlap overall is the overlap between users in the event and PDP logs in the *Asia-B* dataset, which includes 720K users.

If we look at the log types that exist with IMSI information in all datasets — event, summary and PDP logs — the overlap sizes are 110 K, 230 K, 35 K and 180 K for *Asia-A*, *EU*, *North America* and *Asia-B* respectively. For someone looking to correlate different log types with respect to users, such information could be valuable in deciding which datasets to include and for which log types it is feasible.

More detailed results on these overlaps can be found in Appendix D.

Table 5.5: Overlaps between user sets in different log types (Asia-A), ordered by overlap size.

Log types	Overlap size	% of total
Flow, PDP	254869	20.08%
Event, Flow, PDP	218988	17.25%
Event, Flow	218988	17.25%
Event, PDP	218988	17.25%
Flow, Summary, PDP	130575	10.29%
Flow, Summary	130575	10.29%
Summary, PDP	130575	10.29%
Event, Summary	110398	8.70%
Event, Flow, Summary, PDP	110396	8.70%
Event, Flow, Summary	110396	8.70%
Event, Summary, PDP	110396	8.70%

Table 5.6: Overlaps between user sets in different log types (EU), ordered by overlap size.

Log types	Overlap size	% of total
Flow, Summary	524406	62.54%
Summary, PDP	522214	62.28%
Flow, Summary, PDP	521604	62.21%
Flow, PDP	521604	62.21%
Event, PDP	269359	32.13%
Event, Summary	231301	27.59%
Event, Flow, Summary	231092	27.56%
Event, Flow	231092	27.56%
Event, Summary, PDP	230270	27.46%
Event, Flow, Summary, PDP	230099	27.44%
Event, Flow, PDP	230099	27.44%
Summary, Packet	137158	16.36%
Flow, Summary, Packet	137157	16.36%
Flow, Packet	137157	16.36%
PDP, Packet	137024	16.34%
Summary, PDP, Packet	137023	16.34%
Flow, Summary, PDP, Packet	137022	16.34%
Flow, PDP, Packet	137022	16.34%
Event, Flow, Summary, Packet	80644	9.62%
Event, Flow, Packet	80644	9.62%
Event, Summary, Packet	80644	9.62%
Event, Packet	80644	9.62%
Event, Flow, Summary, PDP, Packet	80586	9.61%
Event, Flow, PDP, Packet	80586	9.61%
Event, Summary, PDP, Packet	80586	9.61%
Event, PDP, Packet	80586	9.61%

Table 5.7: Overlaps between user sets in different log types (North America), ordered by overlap size.

Log types	Overlap size	% of total
Event, PDP	318969	16.56%
Flow, Summary	192603	10.00%
Flow, PDP	190953	9.91%
Summary, PDP	190948	9.91%
Flow, Summary, PDP	190941	9.91%
Flow, Summary, Packet	80611	4.19%
Flow, Packet	80611	4.19%
Summary, Packet	80611	4.19%
PDP, Packet	79754	4.14%
Flow, Summary, PDP, Packet	79384	4.12%
Flow, PDP, Packet	79384	4.12%
Summary, PDP, Packet	79384	4.12%
Event, Summary	35633	1.85%
Event, Flow	35629	1.85%
Event, Flow, Summary	35628	1.85%
Event, Summary, PDP	35373	1.84%
Event, Flow, PDP	35370	1.84%
Event, Flow, Summary, PDP	35369	1.84%
Event, Packet	18608	0.97%
Event, Flow, Summary, Packet	18507	0.96%
Event, Flow, Packet	18507	0.96%
Event, Summary, Packet	18507	0.96%
Event, PDP, Packet	18444	0.96%
Event, Flow, Summary, PDP, Packet	18343	0.95%
Event, Flow, PDP, Packet	18343	0.95%
Event, Summary, PDP, Packet	18343	0.95%

Table 5.8: Overlaps between user sets in different log types (Asia-B), ordered by overlap size.

Log types	Overlap size	% of total
Event, PDP	729986	54.83%
Summary, PDP	255927	19.22%
Event, Summary, PDP	180779	13.58%
Event, Summary	180779	13.58%
Summary, PDP, Packet	157161	11.81%
Summary, Packet	157161	11.81%
PDP, Packet	157161	11.81%
Event, Summary, PDP, Packet	117078	8.79%
Event, Summary, Packet	117078	8.79%
Event, PDP, Packet	117078	8.79%
Event, Packet	117078	8.79%

5.1.3 Data integrity of event logs

The gap analysis presented previously in the report gives a basic view into the integrity of the different log recordings. For the radio network event logs, we can also specifically study the effect of a number of factors that affect the integrity of the data, since their occurrences are logged as specific events. Each RNC has a number of processing units that can be individually affected by a number of different error types, presented[2] below with their respective abbreviations used in the figures:

- `FILE_SIZE_EXCEEDED` (abbr. "`>FILE_SIZE`"): The maximum file size for a processing unit, for the current 15 minute logging period was exceeded, after which no other events were logged that period.
- `OVERLOAD`: The processing unit suffered overload, disabling event logging until the load is lower again.
- `PARTIAL_OVERLOAD` (abbr. "`PART_OVLOAD`"): The processing unit suffered partial overload, disabling logging of some event types until the load is lower again.
- `RESTART`: The processing unit was restarted, and event logging is resumed when it is online again.

A Cumulative Distribution Function (CDF) plot over the error rate per processing unit (considering the set of all processing units on all RNCs in each dataset) is presented in Figure 5.7. Note that the X-axis has a logarithmic scale. This was chosen to be able to convey details at different parts of the value range. The fraction of values equal to zero is presented in separate X-axis on the left side of the plot.

The error rate is measured as non-logging time in seconds on a processing unit, per hour. In this way we can compare the error tendencies, irrespective of log time span. Note that data is missing for the *EU* dataset since error events were not logged properly in that recording.

In the plot, we can see that the processing units in the *Asia-A* dataset tend to have much more error time per time unit than the other two datasets, with a median of 0.1 s of error time per hour compared to a median of 0 error time in *Asia-B* and *North America*. Looking at the 90th percentile, we can see that 10% of the processing units in the *Asia-A* dataset have error rates over 4 s/h, while the corresponding number is 0.07 s/h for *Asia-B* and 0.006 s/h for *North America*. We can also see that the maximum error rate for any processing unit goes up to around 500 s/h in *Asia-B* while being around 60 s/h for *Asia-B* and 2 s/h for *North America*. While *Asia-A* is more affected by errors than the other datasets, still only a small part of its processing units were affected by errors, a small fraction of the time. The overall effect of the errors on the studied datasets can therefore be said to be quite small.

We can also consider the percentage of processing units without errors at any point in time, which we will refer to as the logging level. A CDF plot of the fraction of time spent at any logging level up to a certain logging level is presented in Figure 5.8. We can see that only around 1 % of the total time is spent in logging levels below 97% in any of the datasets, and in *Asia-B* and *North America*, less than 2% of the time is spent in a logging level less than 100%. For *Asia-A* we instead see that aorund 30% of the total time is spent in a logging level higher than 97.5% and less than 100%. The varying size of the discrete

[2] There are other possible error triggers, but since they did not appear in any of the studied logs they are omitted here.

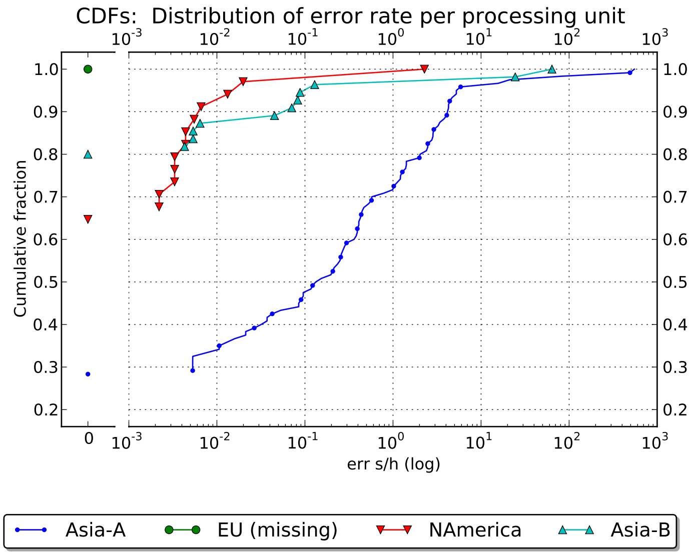

Figure 5.7: Error rate per processing unit in seconds per minute, in the different datasets

steps, in between datasets, is due to the varying total number of processing units within each dataset, which decides the granularity of the fractions.

It is interesting to note that all three datasets in the plot suffered from the recurring 15 minute time gaps, as seen previously in this section, and yet we see no time at logging level 0. If the recurring gap was explained by a low logging level, we would expect to see around 1% of the time at logging level 0, since we observed no events during these gaps, and the time in the gaps was around 1% of the total time (see Table 5.1).

It is also interesting to study the causes for this non-logging time. Pie charts of the triggers of non-logging time in the different datasets are presented in Figure 5.9, with slice size proportional to the total non-logging time attributed to each trigger. We see that in both *North America* and in *Asia-B*, the PARTIAL_OVERLOAD trigger is responsible for all non-logging time while in *Asia-A*, this trigger is only responsible for around 10% of the non-logging time. Instead, the FILE_SIZE_EXCEEDED accounts for around 85% of the non-logging time in *Asia-A*. This indicates some problems in the setup of the logging for this recording, since the file sizes are exceeded often. It is possible that the hardware used in this radio network does not allow for the storage and transfer of the quantities of log data that would be required to more completely record event logs.

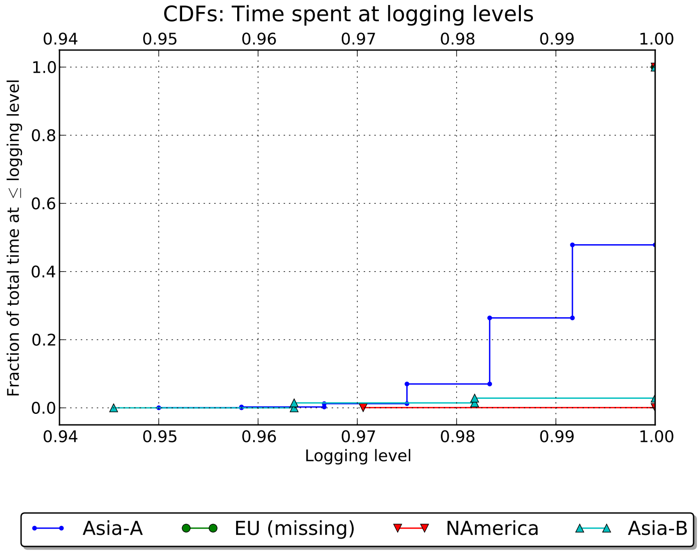

Figure 5.8: A plot over the cumulative time spent in logging levels up to a certain logging level, in the different datasets

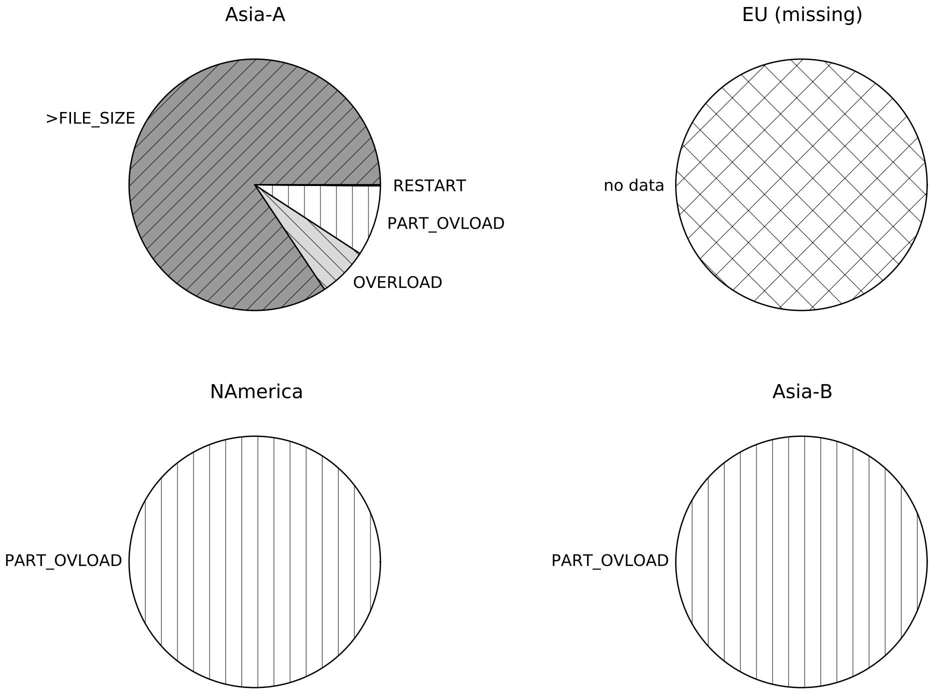

Figure 5.9: Total error time by error trigger

5.2 Basic dataset analysis

The key statistics described in Section 4.4 were collected and the sampled values were saved to allow flexible analysis. For each of the value distributions, two plots were generated for the HTML report: a Cumulative Distribution Function (CDF) plot and a box plot. A table of mean, variance, minimum value, maximum value, median and different percentiles was also assembled for each value distribution, for the HTML report.

While the generated report contains large number of statistics presented from the different perspectives, the key statistics studied in this report are presented as CDF plots since it is a very informative view that still allows for easy comparison between dataset distributions.

A *sequence* in this section refers to a radio network event sequence, as defined in Section 4.2.

5.2.1 Radio network logs

Sequence duration

A CDF plot over sequence duration in the different datasets is presented in Figure 5.10.

We can see that the *Asia-B* dataset has many sequences with long duration. Just below 90% of its sequences are shorter than 100 s, while the corresponding percentage for the other datasets is above 97%. Up to the median the curves for *Asia-B* and *North America* are similar, but from there on *Asia-B* has markedly longer sequences. We can here observe that *North America* has a large group of sequences with duration 10-15 s, covering more than 30% of its sequences. A seemingly small concentration also occurs for *Asia-B* where around 3% of its sequences are around 600 s long, which we will get back to later on in the report. The *EU* and *Asia-A* datasets have similar distributions, and have shorter durations than the other two datasets, up until the steep rise for *North America*.

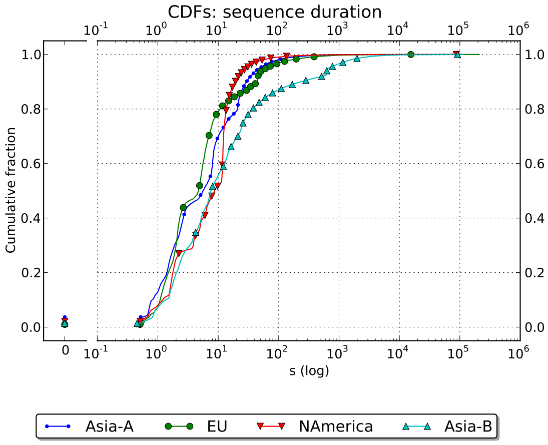

Figure 5.10: CDF:s for sequence duration in the four datasets. The duration is plotted on a logarithmic scale.

Sequence length

A CDF plot over the number of events in the sequences (sequence "length") in each of the four datasets is presented in Figure 5.11. As for the sequence duration, we can see that the *Asia-B* data set has more sequences with higher values. Around 80% of its sequences have less than 10 events, while for the other datasets this fraction is around 95%. Still, we can also see that in *North America* the median is the highest, at 7 events per sequence. Once again, the distributions for *EU* and *Asia-A* are similar. While the median of *North America* is decidedly higher than for *EU/Asia-A*, the curves are similar after the 90th percentile.

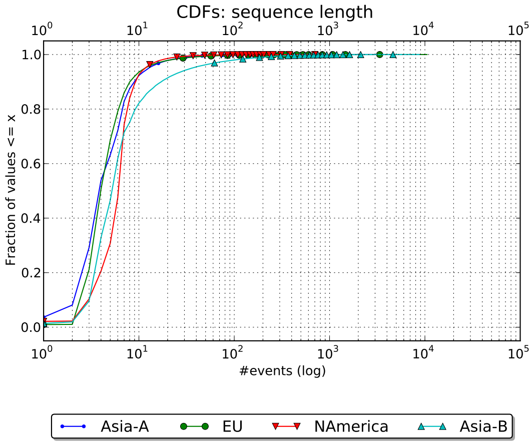

Figure 5.11: CDF:s for the number of events in the sequences in the four datasets. The number of events is plotted on a logarithmic scale.

Number of channel switches per sequence

A CDF plot over the number of channel switch events per sequence in each of the four datasets is presented in Figure 5.12. To start with, we can here see big differences in how many sequences contain a channel switch at all. In *North America* only around 45% of sequences contain no channel switch, for *Asia-B* this is just above 55%, while for *EU/Asia-A* this number is around 80%. We can also note that in *Asia-B* the sequences generally have more channel switches, with 10 % of them having more than five such events, compared to around 3% for the other datasets.

Number of soft handovers per sequence

In contrast to the channel switches, the distributions over the number of soft handovers per sequence are fairly similar. A CDF plot over the number of soft handover events per sequence is presented in Figure 5.13. We can see that in all datasets, around 60% of the sequences lack soft handover events. The *Asia-B* dataset has a slightly higher number of soft handovers, with e.g. around 5 percentage points more sequences with more than 10

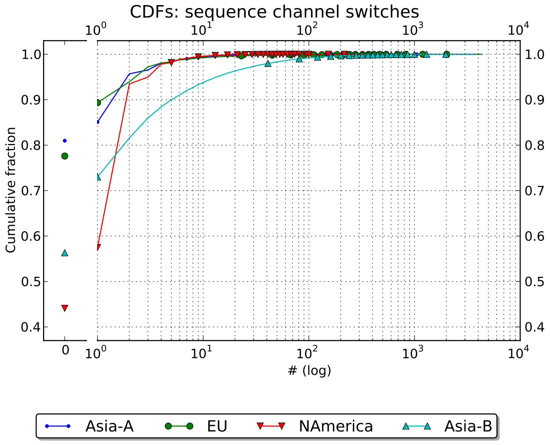

Figure 5.12: CDF:s for the number of channel switches per sequence in the four datasets. The number is plotted on a logarithmic scale.

events. Since *Asia-B* had more sequences with longer duration (Figure 5.10) it could still be that it has a lower frequency of occurrences per time unit.

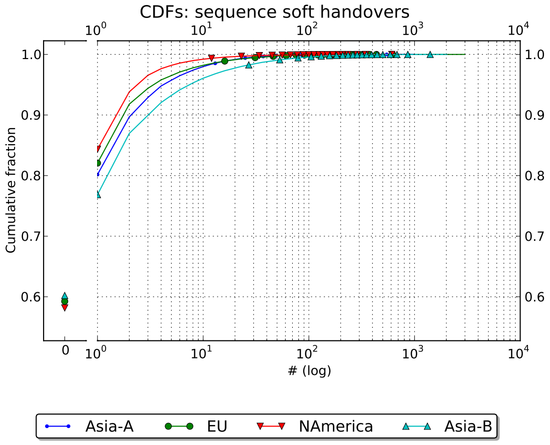

Figure 5.13: CDF:s for the number of soft handovers per sequence in the four datasets. The number is plotted on a logarithmic scale.

Number of cell updates per sequence

For the cell updates we can see that there are some differences in how many sequences contain the event at all, between the datasets. A CDF plot over the number of cell update events per sequence is presented in Figure 5.14. Around 85% of the sequences in the *Asia-A/EU* datasets lack cell updates. This percentage is much smaller in the other two cases: around 60% for *Asia-B* and as small as 45% for *North America*.

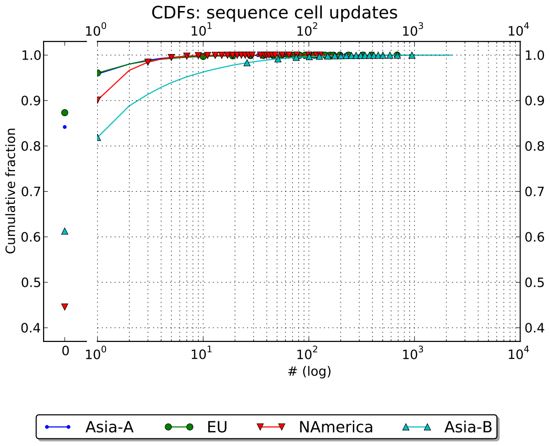

Figure 5.14: CDF:s for the number of cell updates per sequence in the four datasets. The number is plotted on a logarithmic scale.

Number of HS-DSCH cell changes per sequence

A CDF plot over the number of cell update events per sequence is presented in Figure 5.14. We can see that very few sequences contain this event, with the most being in *Asia-B* at around 10% and the other datasets having it appearing in only 2.5-5% of the sequences.

5.2.2 IP logs

There are many common attributes that can be extracted in all three IP log types (Summary-, Flow- and Packet logs) such as the distribution over devices and services. These statistics are only presented and discussed for the Summary log, since they seemed to be similar for the other two log types.

Summary activity traffic volume

Recall from Section 2.2.2 that a summary activity covers the traffic of one application for a user during one minute. A CDF plot over total traffic volume in bytes (uplink+downlink) per summary activity for the four datasets is presented in Figure 5.16.

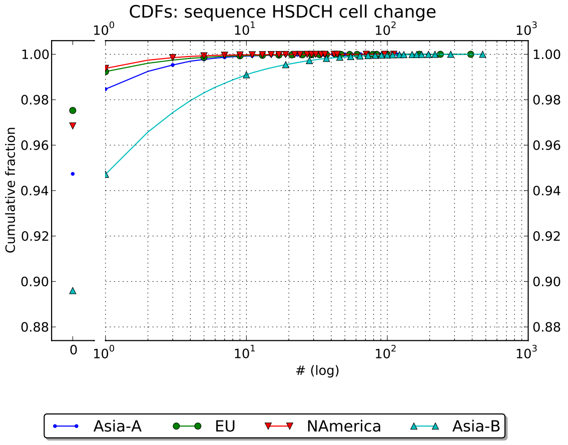

Figure 5.15: CDF:s for the number of HS-DSCH cell changes per sequence in the four datasets. The number is plotted on a logarithmic scale.

We can see that summary activities tend to have more traffic in *Asia-A*. For each value between the median at 10^3 bytes up to the 90th percentile at 10^5, *Asia-A* consistently has 5-10 percentage points more of its sequences over that value.

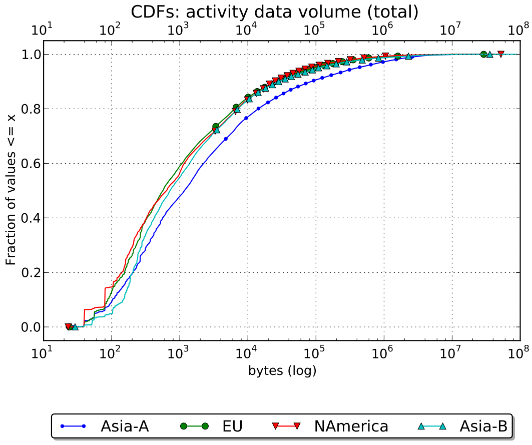

Figure 5.16: CDF:s for the total traffic volume (uplink + downlink) during each summary activity for the four different datasets. The volume is plotted on a logarithmic scale.

Summary traffic volume per user

For this statistic the total traffic on both uplink and downlink was tracked per IMSI. The results are presented in Figure 5.17. We can see that the users in the *Asia-B* dataset have a much higher traffic volume. The median user in *Asia-B* had a traffic volume of 10^8 bytes (100MB) while the medians in the other data sets are 3×10^5 to 2×10^6 (0.3 to 2 MB).

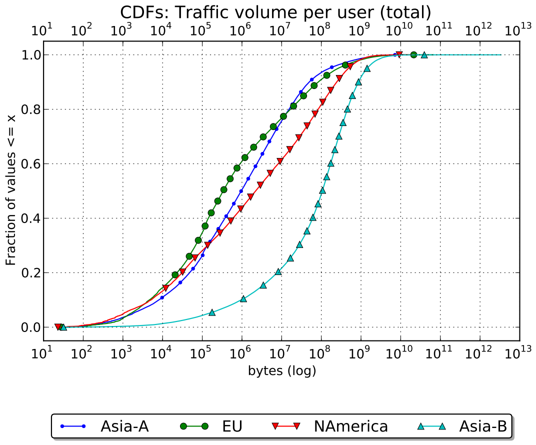

Figure 5.17: CDF:s for the total traffic volume (uplink + downlink) per user for the four different datasets. The volume is plotted on a logarithmic scale.

Summary traffic volume per device type

The traffic volume distribution over device types is presented in a pie chart in Figure 5.18. There are five types: handheld, PC, tablets, routers and Machine to Machine (M2M). All slices after 90% of the total are collapsed into one slice, labeled 'other'. We can see that PCs dominate the traffic volume in *Asia-A* and *EU*, while handhelds are responsible for most of the traffic in *Asia-B* and *North America*. This indicates a clear difference between datasets as to which device types are generating the most traffic.

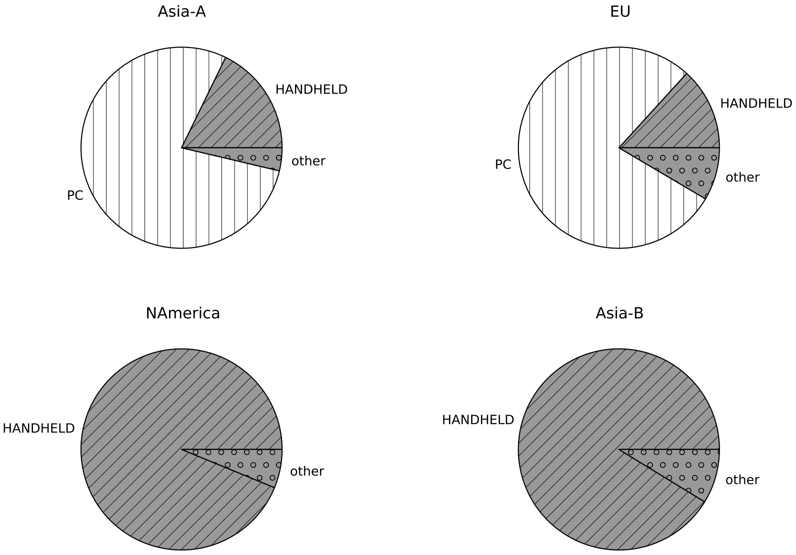

Figure 5.18: Total traffic volume distribution (uplink+downlink) per device type

Summary traffic volume per device Operating System (OS)

The traffic volume distribution over device OS is presented in a pie chart in Figure 5.19. Unfortunately the mapping to device OS was not known for a large portion of the devices in *Asia-A/EU*. The correspondence between Unknown in this plot and PC in Figure 5.18 suggests that the OS mapping is missing for PCs.

Nevertheless, we can see that Android dominates the traffic volume in *North America* at around 80% of the total, and about 60% of the total in *Asia-B*, which also has a significant portion (around 25%) from iOS devices. Only *Asia-A* and *EU* have similar distributions, while there are clear differences in between the others.

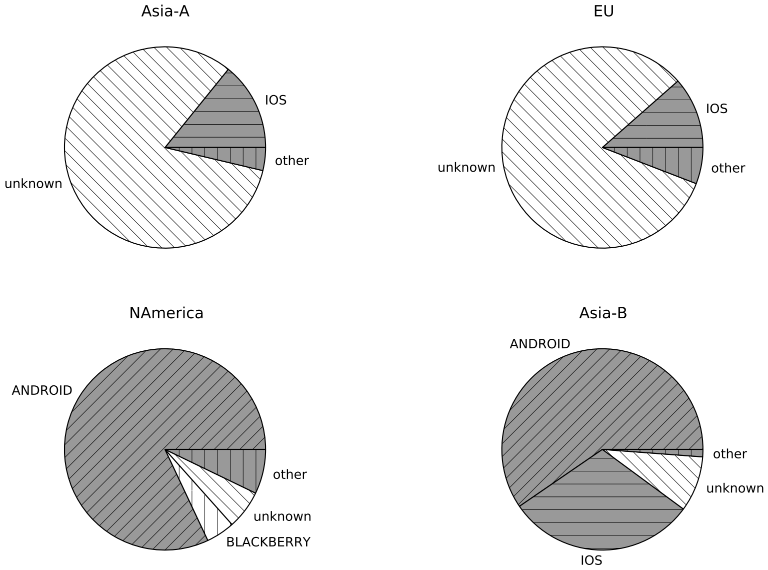

Figure 5.19: Total traffic volume distribution (uplink+downlink) per device OS

User distribution over device type

The user distribution over device types is presented in a pie chart in Figure 5.20. We can see that the handheld type dominates in all datasets. The only other category that appears with a significant amount is PC, which around 10% of users have in the *Asia-A* and *EU* datasets. It is interesting to note that while only 10% of users have PCs in these datasets, they are responsible for more than 75% of the traffic volume, as could be seen in Figure 5.18. We see that the distributions are similar for *Asia-A/EU*, while *Asia-B* has a higher portion of handheld devices, and *North America* almost exclusively has handheld devices.

User distribution over device OS

The user distribution over device OS is presented in a pie chart in Figure 5.20. We can see that there are clear differences in the device fleets between the datasets. *Asia-A* is dominated by iOS (IPhones/IPads) while *North America* is dominated by Android based phones. It is also the only dataset where a significant fraction of the users (around 20%) use BlackBerry phones.

The *EU* dataset seems to be dominated by proprietary and Symbian phones, and could be labeled a feature phone [3] heavy network. On the contrary, the *Asia-A* dataset is dominated by Android and iOS phones, and can be considered a smartphone heavy network.

[3] "Dumb" phones, or at least "not-as-smart-as-Android/iOS-phones"

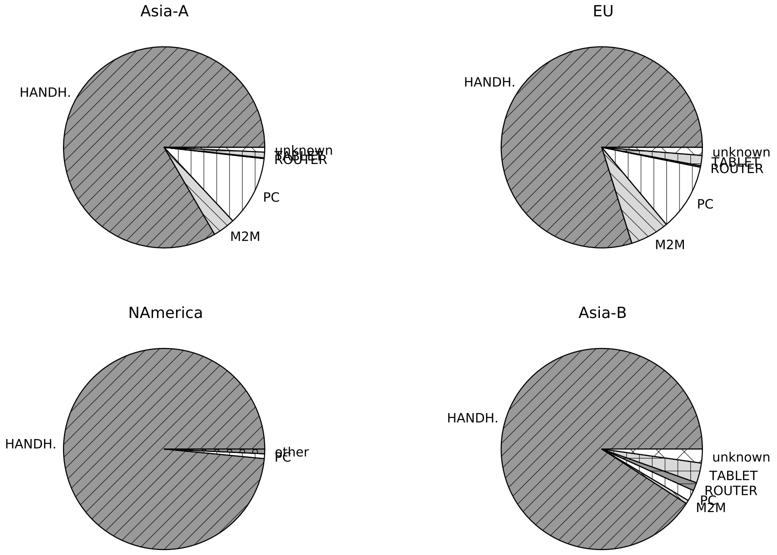

Figure 5.20: The user distribution over device types

In general we see that there are large differences between the distributions in the different datasets.

Flow duration

Recall the term flow from Section 2.2.2. A CDF plot over the flow duration in the different datasets is presented in Figure 5.22. We can see that *Asia-A* has longer flows up until the 60th percentile at around 3 seconds, and after this the flow duration is similarly distributed in all of the datasets.

Traffic volume per flow

A CDF plot over the total traffic volume (uplink + downlink) in the different datasets is presented in Figure 5.23. We can see that the flows in *Asia-B* and *North America* tend to have a higher traffic volume. The 75th percentile is 6 Kb while for *Asia-A/EU* it is instead around 1-2 Kb.

Packet size

A CDF plot over UDP/TCP packet sizes in the different datasets is presented in Figure 5.24. The minimum size for a TCP packet is 40 bytes and the Maximum Transmission Unit is 1500 bytes which corresponds well with the sharp jumps in the start and end of the plot. We can see that the packets in the *Asia-B* dataset seem to be slightly bigger than in the

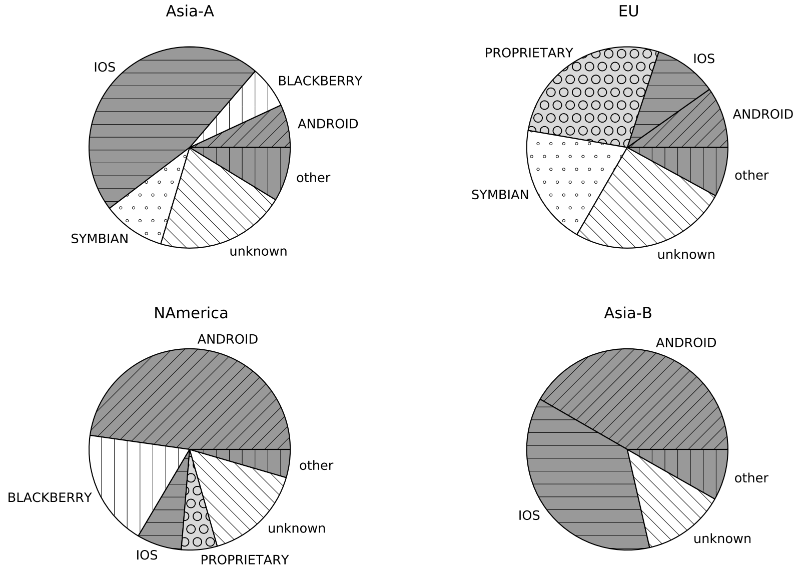

Figure 5.21: The user distribution over device OS

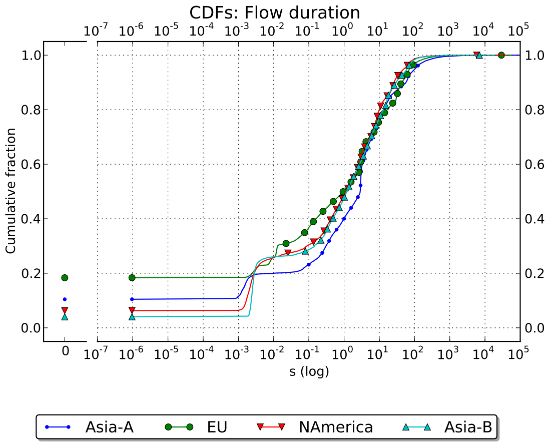

Figure 5.22: CDF:s for the number of flow duration in the four datasets. The duration is plotted on a logarithmic scale.

other data sets, but the differences are not that large. Unfortunately, the data for *Asia-A* is unavailable.

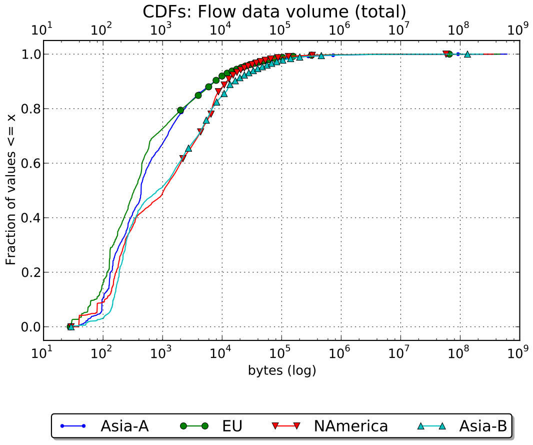

Figure 5.23: CDF:s for the total traffic volume during flows in the four datasets. The volume is plotted on a logarithmic scale.

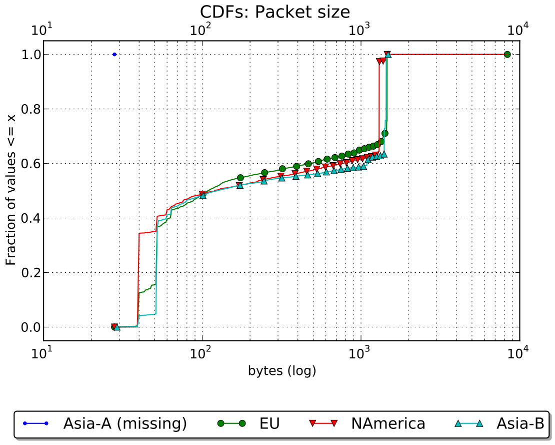

Figure 5.24: CDF:s for the packet size in the four datasets. The size is plotted on a logarithmic scale.

5.3 Analysis of common event sequences

5.3.1 Determining K

A prerequisite to running the cluster analysis is to decide how many clusters to use. The Gap-statistic (see Section 3.2.3) was run for $K = 1..20$ clusters, with the number of tested randomly generated distributions, $B = 30$. Unfortunately, it was infeasible to run on the full dataset because of time constraints, and therefore only 50 000 randomly sampled sequences from each dataset was used for this analysis.

The plot of the achieved gaps for different K is presented in Figure 5.25. The error bars show s_k, the adjusted standard deviation of the error on the randomly generated datasets (see Section 3.2.3 for details). Since the error bars are quite large compared to the deviance from 0 we see that there is a quite large variance in the error for the randomly generated datasets.

While it is impossible to tell from the graph, the actual result of the gap statistic method is that $K = 1$ should be used, since:

$$Gap(1) = 0 > Gap(2) - s_2 = 9 \times 10^{-16} - 1 \times 1^{-14} \approx -1 \times 10^{-14}$$

Recall from Section 3.2.3 that the gap statistic method is specifically designed to identify the appropriate number of clusters to use when there are well separated clusters in the data. One conclusion we can draw from this is that there is no strong evidence for the existence of well separated clusters in the data given the used distance metric.

The analysis done in this project is however not deemed to require that the clusters are very well separated. The clustering will still provide a useful discretization of the space of sequences that we can use to to analyze differences in how common certain types of sequences are in the different datasets.

The gap statistic curve has a maximum at $K = 11$, while also having a slightly smaller error bar at this point, which is a good indication that there are somewhat well separated clusters when $K = 11$ is used. Because of this, $K = 11$ clusters were used in the cluster analysis.

5.3.2 Cluster perspectives

Overview

The K-means clustering algorithm was run as described in Section 4.5.3, using 1 million randomly sampled sequences from each dataset as the input data. Using the motivation from the previous section, $K = 11$ clusters was used. The found clusters were of varying size and had different properties. An overview of the clusters is shown in Table 5.9, which includes the relative size of the clusters and a relative feature value marker of 1-4 bullets for each feature for each cluster. To refer to the clusters, the 11 indices 0-10 are used. The indices were picked in order of the median duration of each cluster. The short names used for the features are described in Table 4.3.

The markers in the table indicate the relative feature value of a cluster center compared to the other clusters. These were generated by using the feature values (after the normalization and transformation described in 4.5.3) for each respective cluster center. The maximum and minimum feature values for the cluster centers were used to scale the values to the range 0..1, after which the values were categorized into four ranges, with $[0, 0.25)$ corresponding to one bullet and $[0.75, 1.0]$ corresponding to four bullets. In this way, one

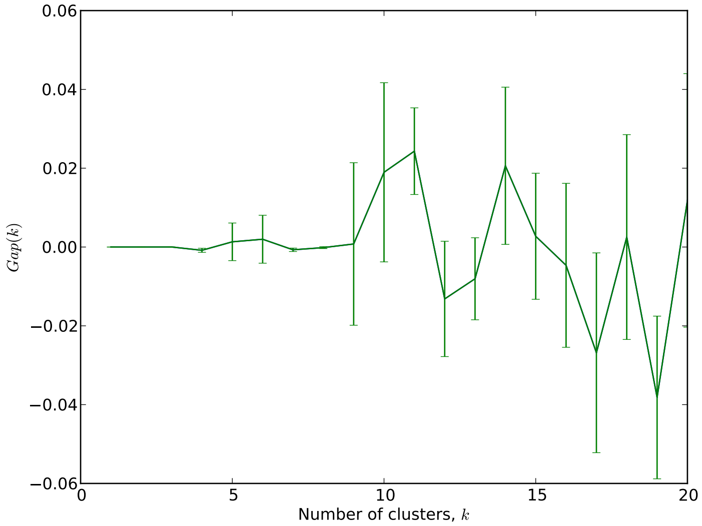

Figure 5.25: The value of the gap statistic for different number of clusters used (K). The error bars show s_k, the adjusted standard deviation of the error on the randomly generated datasets (see Section 3.2.3 for details).

bullet indicates a low relative value for that feature compared to the other cluster centers, and four indicates a high relative value.

As we can see in the table, almost all clusters are different in terms of their feature properties. Only clusters 1 and 3 have the same relative feature values using this crude discretization. It is also apparent that most sequences are members of clusters with a low relative duration, with clusters 0-5 covering 82.3% of all sequences.

Sampled sequences comparison plot

While the overview presented in the last section gives some idea about the sequences in each cluster, it does not describe them in any detail. To get a better view what kind of sequences appear in the clusters, example sequences from each cluster are visualized in Figure 5.26. In this figure, six sequences from each cluster are visualized to show how their events occur in time. The five bottom-most for each cluster are randomly sampled from all sequences in the cluster, while the top one is the sequence which is closest to the cluster center. The key events that occur in the sequences are visualized with different kinds of markers, as described in the legend. To cover sequences of very varying length, the time scale is logarithmic. While this blurs together events in the later stages of long sequences, it is still possible to compare the length of the sequences, and the rate at which events occur.

We can see that cluster 0 seems to exclusively contain sequences that only have an RRC connection request event. As was indicated in the previous section, clusters 1 and 3 have similar events, but here we also see that the duration is longer in cluster 3.

Table 5.9: Cluster overview

Idx	Cluster size	duration	#chsw	#soho	#cellupd	#hs-cc
0	2.1 %	●	●	●	●	●
1	20.8 %	●●	●	●	●	●
2	14.2 %	●●	●	●●	●	●
3	19.2 %	●●	●	●	●	●
4	10.2 %	●●	●	●●	●	●
5	15.8 %	●●●	●●	●	●	●
6	9.1 %	●●●	●●	●●	●	●
7	2.6 %	●●●	●	●●●	●	●
8	2.5 %	●●●	●●●	●●●	●●	●●
9	2.7 %	●●●●	●●●	●	●	●●●
10	0.8 %	●●●●	●●●●	●●●●	●●●●	●●●●

It is clear that all clusters differ with respect to some property. Sequences in clusters 5 and 6 e.g. seem to have similar duration, but in 6 they are distinguished by having more soft handovers and HS-DSCH cell changes. Similarly, clusters 7 and 8 differ in that there are more channel switches for the sequences in 8.

Cluster 9 has some especially distinguishing properties. Its sequences have a very long duration, and contain almost exclusively channel switches and cell updates. In contrast, the two other clusters with similar duration, clusters 8 and 10, contain soft handovers and HS-DSCH cell changes at a quite high rate.

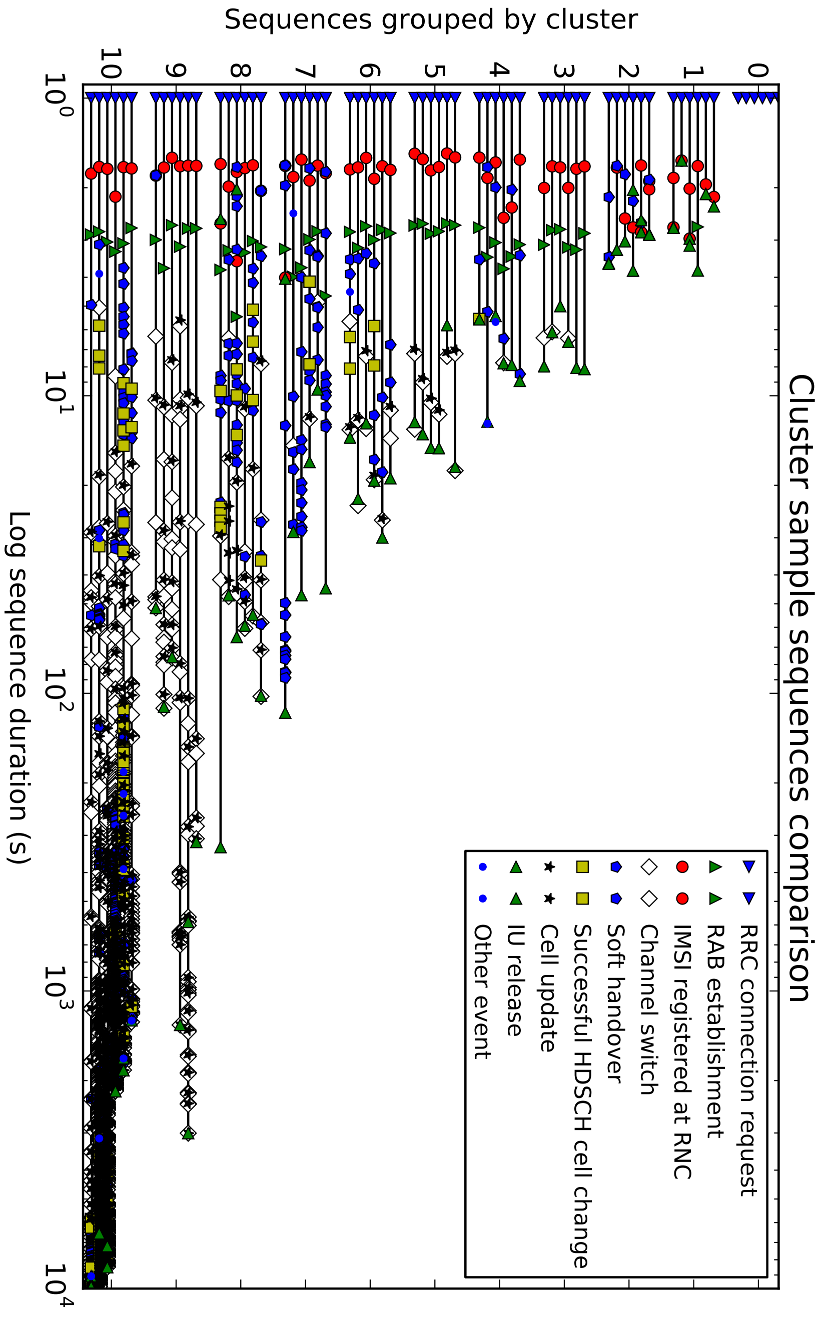

Figure 5.26: Example sequences from each cluster.

Cluster distribution comparison

To understand the clusters on a deeper level we can study Cumulative Distribution Function plots for different properties.

Duration In Figure 5.27, we can see CDF:s for the sequence duration in the different clusters. While we can see that many clusters overlap in the durations covered, some clusters have almost identical duration distribution, such as 3/4, and 5/6. It is to be expected that clusters overlap in a feature like this, since many features have been used for finding the clusters, and the other features could be distinguishing sequences of the same length, which we saw in the previous section. As mentioned before, the cluster indices were chosen in the order of median sequence duration. This will later give us a simple tool in relating CDF:s for other properties to sequence duration by just looking at the index order.

An interesting thing to note in the figure is the steep slope in the curve for cluster 9 at around 600 seconds. Recall that two steep slopes were discovered in the per dataset sequence duration CDF plot in Figure 5.10: for *North America* at around 10-15 seconds and for *ASIA-B* at 600 s. The 600 s bump therefore seems to have ended up in Cluster 9, although we will get back to actually verifying the origin datasets of Cluster 9 later in the report. Judging from the figure, the 10-15 s bump seems to have ended up in clusters 5 and 6, and we will also try to verify later by considering dataset origin distribution in these clusters.

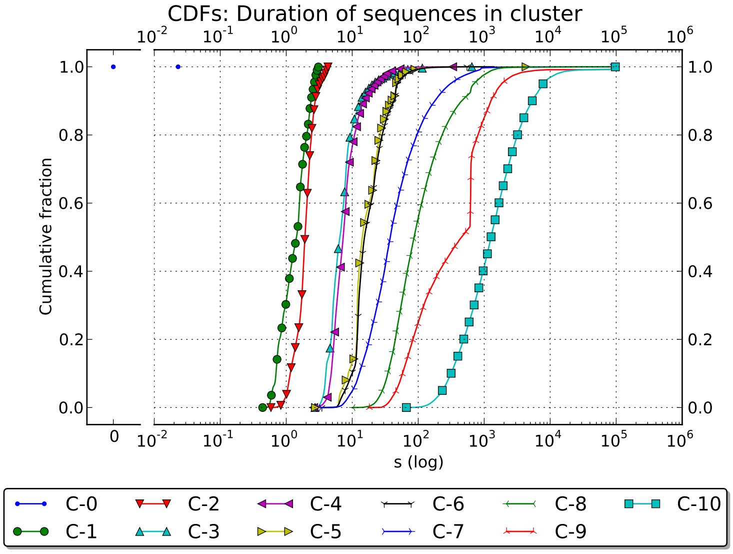

Figure 5.27: CDF:s for the sequence duration in different clusters. The duration is plotted on a logarithmic scale.

Length In Figure 5.28, we can see CDF:s for the sequence length (total number of events) in the different clusters. We can see that Cluster 8 which has a lower median sequence duration than Cluster 9, still has sequences of longer length. Similarly, Cluster

3 has slightly fewer events in its sequences than Cluster 2, while having higher median duration. The distribution in this figure ties in well with the visualization in Figure 5.26, where e.g. the long duration and comparatively low number of events for Cluster 9 gives its example sequence a much less dense character than clusters 8 and 10.

It is worth noting that the sequence length in itself was not used as a basis for any feature for the clustering procedure. Only the sequence duration and specific counts of channel switches, soft handovers, cell updates and HS-DSCH cell changes were used as raw features, and therefore have a direct effect on the clusters found. Of course, the different event counts affect sequence length, so there is an indirect correlation.

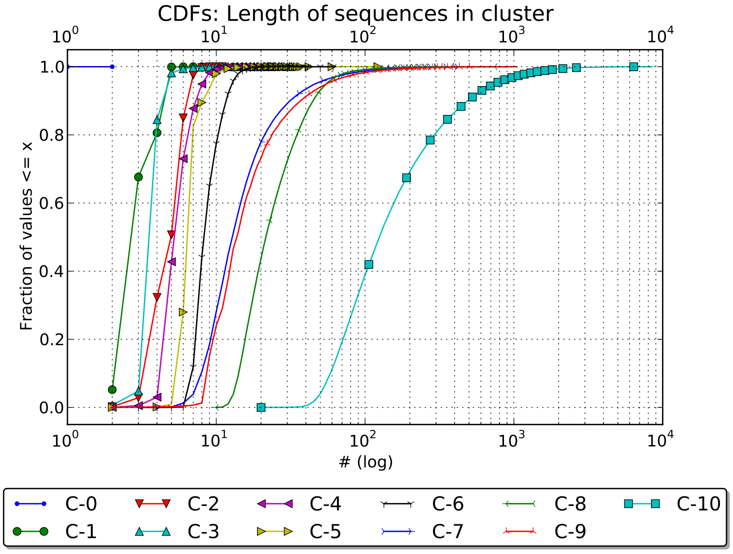

Figure 5.28: CDF:s for the total number of events per sequence in different clusters. The number is plotted on a logarithmic scale.

Channel switches We have seen that some clusters overlap in sequence duration which was the basis for one clustering feature. Next we will look at the counts of the four specific events which were the basis for the other four features used for the clustering. This will present the differences between clusters at a deeper level of detail than what could be seen in the overview in Table 5.9.

In Figure 5.29, we can see the CDF:s for the number of channel switch events per sequence in the different clusters. To start with, we can see that clusters 0, 1 and 2 have no channel switches in their sequences. Also for clusters 3, 4 and 7 the majority (70-90%) of sequences have no channel switches. Clusters 5 and 6 have an almost identical distribution, with around 30% of the sequences having one channel switch, and almost 60% having two, with very few having much more than that. Similarly, clusters 8 and 9 have curves that follow very closely, with a median around 6 switches, with the sequences in Cluster 9 having slighly more sequences with more than 10 switches. Cluster 10 dominates this chart, having the median number of switches being around 50.

Soft handovers A plot of the CDF:s for the number of soft handovers per sequence in the different clusters is presented in Figure 5.30. Here we can see that clusters 0, 1, 3 and 5 have no soft handovers at all. It is interesting to note that Cluster 9, which has a long median duration has the fewest soft handovers of the rest of the clusters. Over 70% of its sequences have no soft handovers. The sequences in clusters 2, 4 and 6 have a majority (55-70%) of their sequences with one soft handover, and no sequence with more than 10 soft handovers.

Another interesting observation is that the sequences in Cluster 7 tend to have more soft handovers than the ones in Cluster 8, even though Cluster 8 has a higher median duration and sequence length which we could se in Figure 5.27 and 5.28.

Once again Cluster 10 clearly has the most number of soft handovers per sequence, with a median of almost 30 compared to 7 for Cluster 7.

Cell updates A plot of the CDF:s for the number of cell updates per sequence in the different clusters is presented in Figure 5.31. We can see that clusters 0-4 have practically no sequences with cell updates. For Cluster 7, almost 90% of the sequences contain no cell updates, while the last 10% contain one. In clusters 5 and 6, 80-90% of the sequences contain one soft handover, 99% contain up to three cell updates.

For clusters 8 and 9, the distributions look very similar, having a median of 3 cell updates per sequence, with Cluster 9 having slightly more sequences with more than 10 cell updates.

Again, Cluster 10 clearly dominates this statistic, having a median of around 25 cell updates per sequence.

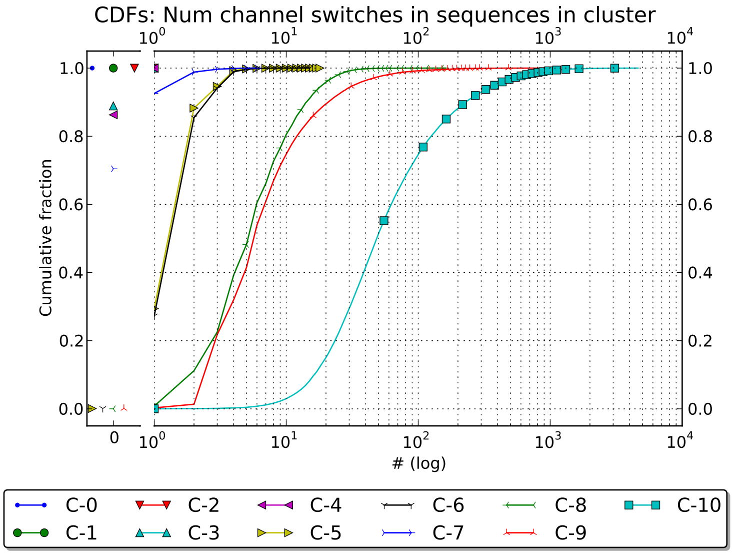

Figure 5.29: CDF:s for the sequence number of channel switch events per sequence in different clusters. The number is plotted on a logarithmic scale.

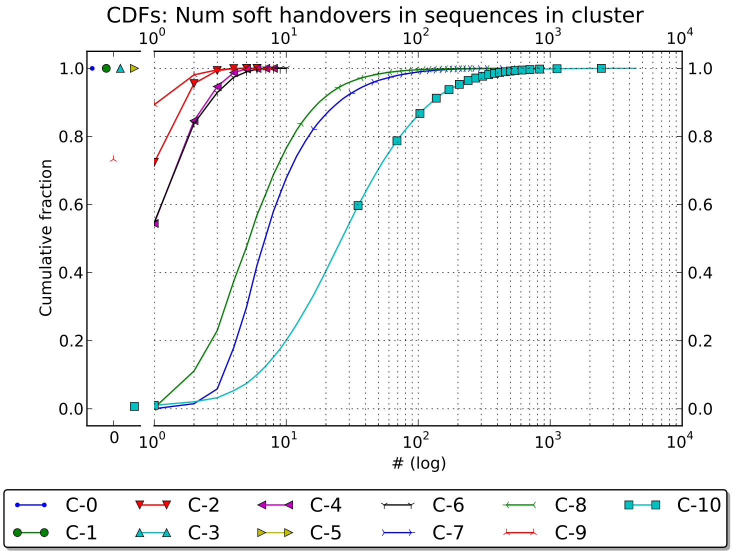

Figure 5.30: CDF:s for the number of soft handover events per sequence in different clusters. The number is plotted on a logarithmic scale.

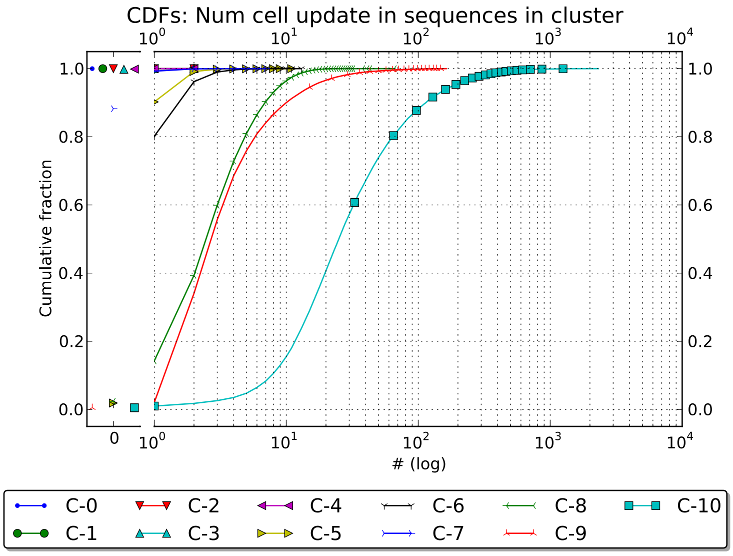

Figure 5.31: CDF:s for the number of cell update events per sequence in different clusters. The number is plotted on a logarithmic scale.

HS-DSCH cell changes A plot of the CDF:s for the number of HS-DSCH cell changes per sequence in the different clusters is presented in Figure 5.32. We can see that no sequences in clusters 0, 1, 2, 3 and 5 have any HS-DSCH cell changes. In clusters 4 and 6, around 90% of the sequences have no cell changes. Furthermore, Cluster 9 has very few sequences (less than 5%) which contain HS-DSCH cell changes.

Cluster 10 also clearly has the most HS-DSCH cell changes, and is the only cluster with a significant number of sequences with more than 10 cell changes. Clusters 7 and 8 have 15%/30% respectively of their sequences with only one cell change, and 15%/30% of their sequence with 2-10 cell changes.

Compared to the previously discussed event distributions per cluster, a difference for HS-DSCH cell changes is that only in Cluster 10 does a clear majority of the sequences contain the event, while for the other event types there has been three clusters where almost all sequences contained the event.

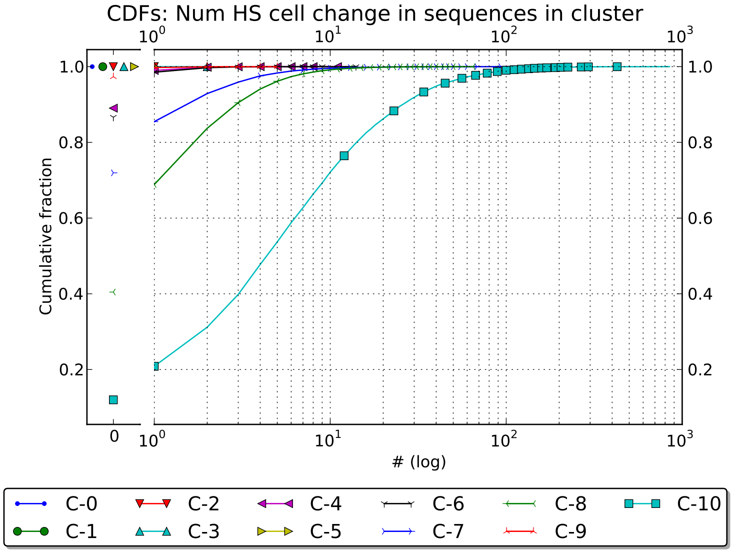

Figure 5.32: CDF:s for the number of HS-DSCH cell change events per sequence in different clusters. The number is plotted on a logarithmic scale.

Device type Similarly as with the sequence length, the type of devices that have generated the sequences in each cluster have no direct effect on the clustering procedure. In fact, while sequence length has an obvious indirect relation to the raw event count features, there is an even less direct relation with device type to the raw features. We can speculate in there being some links though, e.g. that less mobile devices such as the PC device type[4] will have few mobility events such as soft handovers and cell updates.

Pie charts over the distribution over device types per cluster are presented in Figure 5.33. In the bottom-right corner the overall distribution device type in the sequences is

[4] includes laptops

included. We can see that only clusters 8, 9 and 10 have a significant amount sequences generated by PC devices. Recall that the indexing scheme implies that these clusters have the three highest median durations.

We can also note that clusters 1, 2, 3, 4 and 7 have a slightly lower fraction (less than 5%) of sequences generated by non-handheld devices than the remaining clusters. In contrast, clusters 9 and 10 have more than 10% of their sequences generated by non-handheld devices.

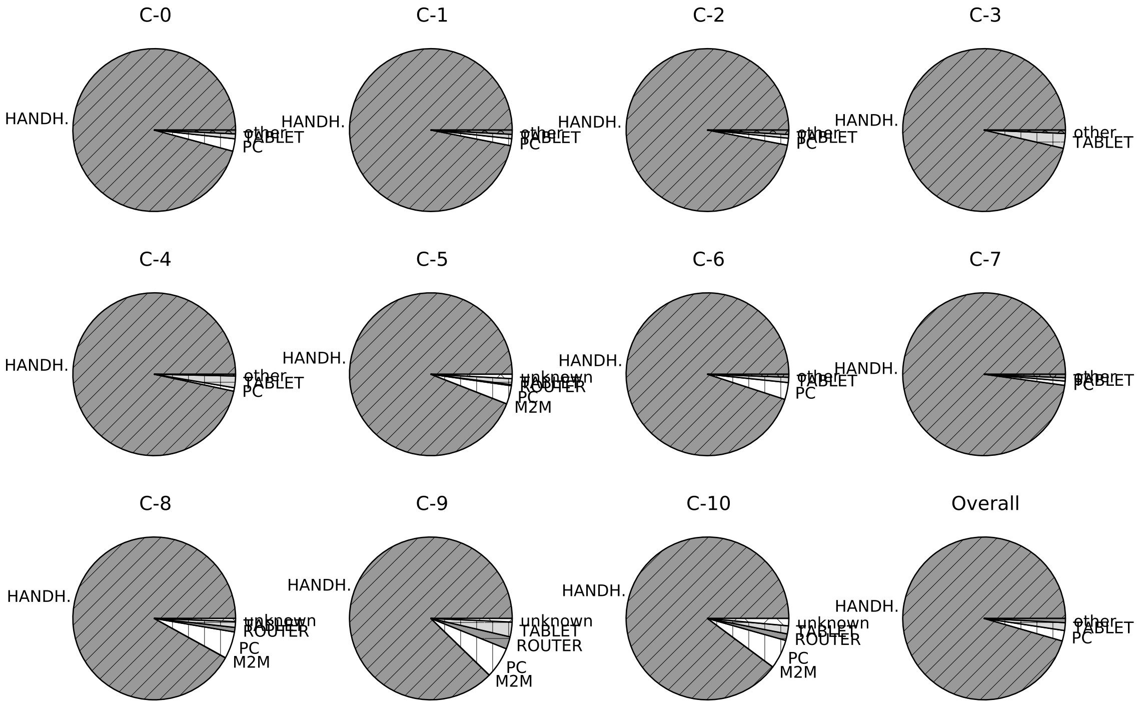

Figure 5.33: Distribution over device type per cluster

Device OS The device OS similarly has no obvious direct relation to the raw features used by the clustering. We can also here speculate in there being some indirect links since the device OS often implies a device type and possibly some hardware features of the phone.

Pie charts over the distribution over device OS's per cluster are presented in Figure 5.33. Note that the bottom right pie chart shows the overall distribution of device OS per sequence. We can see certain deviances from the overall distribution. For one, Symbian is responsible for generating a comparatively high portion of the sequences in clusters 0, 1, 2 and 7. Similarly, while Android is big in the overall distribution, it is even more dominating in clusters 5 and 6. iOS has a bigger portion than overall in clusters 3, 4, 7, 8 and 10 but most notably in Cluster 9 where it accounts for about half of the sequences, compared to 25% in the overall distribution. Blackberry accounts for a notably low portion of the sequences in clusters 7 and 10, but it is otherwise fairly evenly distributed over the clusters.

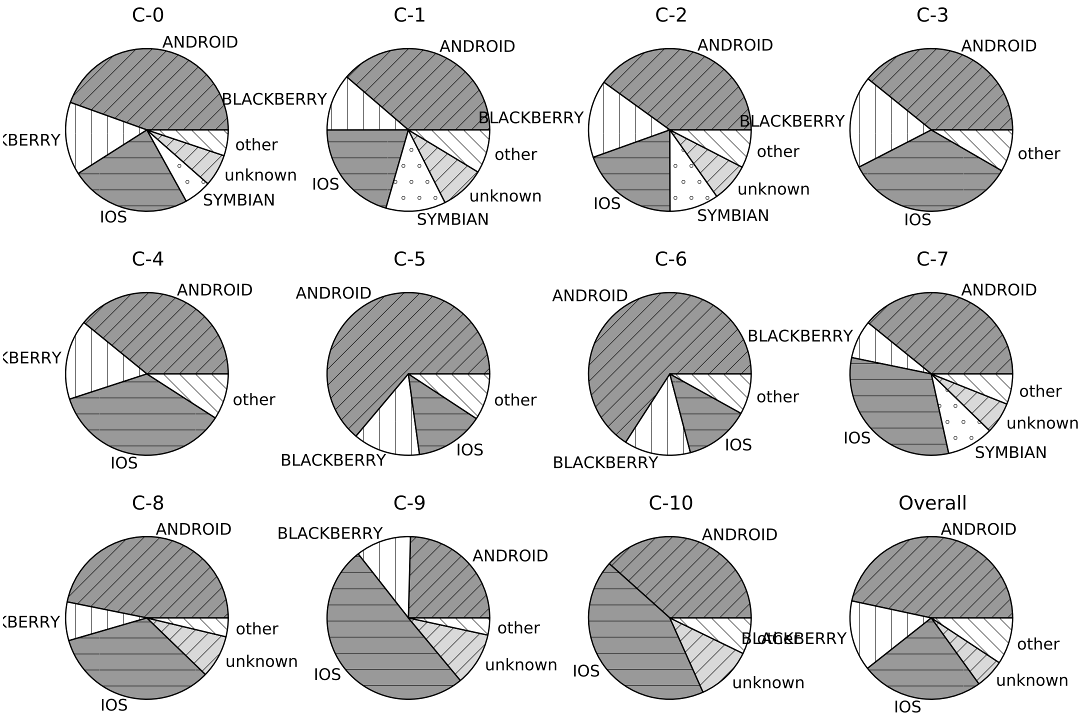

Figure 5.34: Distribution over device OS per cluster

Dataset distribution While we so far have been studying results relating to what properties the different clusters have, the key idea with clustering sequences from different datasets has been to see if there are imbalances in how the sequences from different datasets distribute over the clusters. A pie chart plot over this distribution is presented in Figure 5.35. Recall that the sequences were selected so that the overall distribution has an equal portion from each dataset.

One interesting thing to note in the plot is that clusters 5 and 6 are clearly dominated by sequences from the *North America* dataset. Recall from the plots over sequence duration per dataset (Figure 5.10), and per cluster (Figure 5.27) that we identified a large portion of the sequences from *North America* as having a duration of around 10-15 s, and therefore most probably belonging to clusters 5 and 6. The dominance of *North America* in these clusters further suggests that this group of sequences with similar duration belongs to these clusters.

Another thing that is clear in the plot is that *Asia-B* has a clear dominance in clusters 9 and 10. It may not be that surprising since we saw earlier in Figure 5.10 that there were much more sequences in *Asia-B* with long duration than in the other datasets. We also before identified a small group of sequences with very similar durations around 600 s, that we found strong indications for that they ended up in cluster 9 (from 5.27). Since *Asia-B* dominates this cluster, we have further evidence supporting that this group of sequences ended up in Cluster 9. In this case one could argue that it is even more interesting, since there are several clusters that contain sequences with duration around 600 s, but we have specifically found evidence that they ended up in Cluster 9.

Furthermore we can observe that the *EU* and *ASIA-A* datasets account for a notably high portion of the sequences in clusters 1-4 and 7. *Asia-A* is dominating Cluster 0, with 40% of the sequences. *North America* has a markedly low portion of the sequences in clusters 3 and 7.

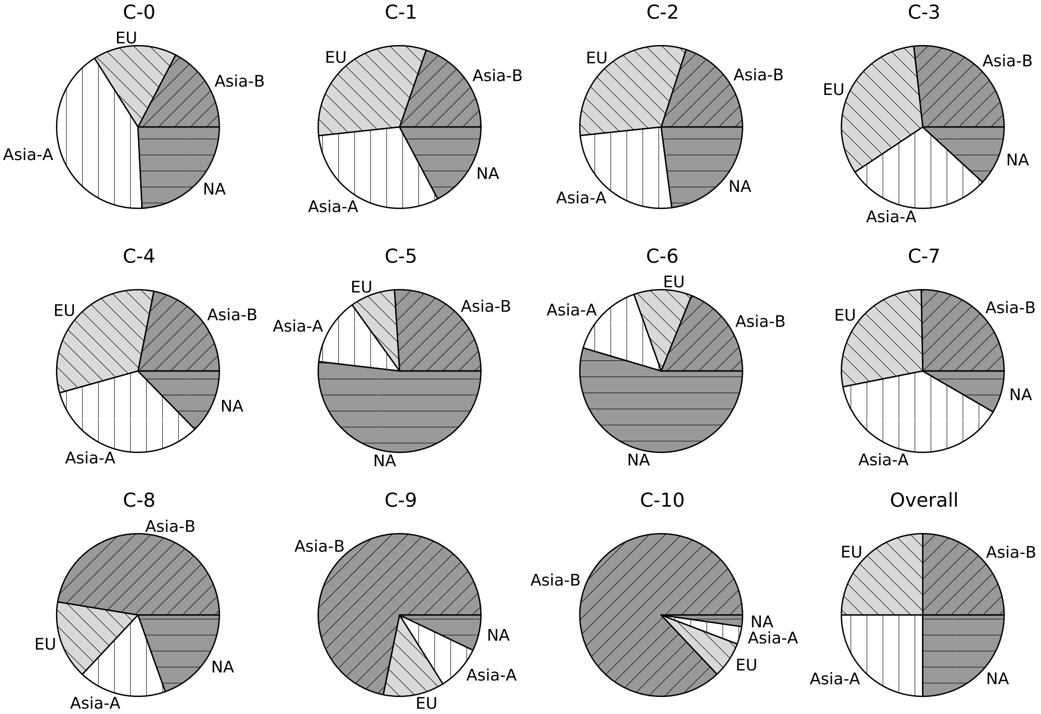

Figure 5.35: Distribution of over datasets per cluster

Table 5.10: Event pattern homogeneity

Rank	Cluster idx	Homogeneity	Count	Median length
1	0	1.000	85299	1
2	4	0.922	708785	6
3	1	0.696	579218	4
4	6	0.638	524622	9
5	5	0.531	175728	7
6	2	0.369	150952	5
7	9	0.178	21648	14
8	3	0.153	38529	4
9	7	0.138	32402	13
10	8	0.093	8702	22
11	10	0.015	736	125

Common event patterns

While it can be informative to consider the duration of sequences, and visualize the occurrences of events in time, another perspective is to simply consider the order of the events in a sequence. To do this, all sequences with a certain event order was counted, and these patterns were then ordered after the number of occurrences of each. The resulting top 3 most occurring event orders can be found in Appendix B.

An overview of the totals for the top 3 event patterns per cluster is presented in Table 5.10, along with median sequence length for comparison. In this table, *Homogeneity* is the fraction of sequences within the cluster that matches one of the top 3 patterns seen in the cluster. We can start by noting that Cluster 0 has homogeneity 1.0 which, by looking at the example sequences in Figure 5.26, seems to be explained by all sequences simply consisting of a single RRC Connection Request event, i.e. having the same event pattern.

It is interesting to note the cases when a cluster has a higher homogeneity than clusters with lower median length, since if we assume completely random transitions between events, we should find fewer examples of sequences that contain more events, and thus a lower homogeneity. In the table we find that clusters 4, 6, 9 and to a degree cluster 7 meets the criteria of having a comparatively high homogeneity in relation to their median sequence length. This indicates that there are certain event patterns in these clusters that are more typical than we could expect for a pattern of that length.

Chapter 6

Discussion

In this chapter the results from the previous section are further analyzed and discussed. Section 6.1 reviews the results regarding time span, log gaps, user coverage and data integrity; and the relevance of such results to a potential analyst. In Section 6.2, the similarities and differences of the studied datasets are examined. The properties of the different clusters are analyzed in Section 6.3 and the differences in how the datasets distributed over these clusters is discussed in Section 6.4. The clustering method itself is evaluated in Section 6.4. Finally, a discussion around the contribution of the common sequence analysis is held in Section 6.6.

6.1 Dataset metadata

6.1.1 Time-wise log coverage

From the time span and gap analysis results from Section 5.1.1, we saw that most log types covered similar time spans across datasets, with the possible exception of the Packet logs, where the *EU* amd *North America* datasets had a total of 2 hour active log time compared to the single hour noted for *Asia-B*, which of course is a large relative difference. Such data is helpful when we interpret some of the basic statistics. While e.g. the large difference in total traffic volume per user (see Figure 5.16) between the *Asia-B* and the other datasets could have been partly explained by the recording time span being significantly longer in that dataset, instead we could here see that the Summary log time span of the *Asia-B* dataset was the shortest among the datasets (see Table 5.2). It is important to be aware of cases where a difference in time span between dataset recordings could explain differences in some statistic, which data such as the referenced table can shed light on.

We could also observe specific downtimes in the logs, such as the logging on an RNC in the *Asia-A* dataset going down in long periods, and a 9 h long gap in the flow logs. It is important to be aware of such issues if you e.g. want to plot a comparison over a statistic over time.

The gap analysis also brought to light a phenomenon in the event logs where, in three out of four logs, there was a regularly occurring 15 minute gap consistently every 24 hours. There seems to be no natural explanation why gaps like this should be observed, and the analysis of the error events did not show any downtimes in the logging in the same magnitude that these recurring gaps constitute. There could have occurred a problem at some step of the processing the data has gone through[1].

Effect of event log gaps on statistical analysis

We can try to reason about the effect of the observed gaps on the measured statistics and clustering results. Since the recurring 15 minute gaps accounted for the majority of all gaps they are most interesting to study. The reasoning here will focus on the integrity of the collected event sequences, since many of the statistics are counted per event sequence, and the sequences also are important in the clustering. The assumption is here that we were supposed to observe events during the gaps, but could not because of some problem with the recording.

It seems reasonable to assume that the *complete* sequences we miss within such 15 minute gaps look more or less the same as the ones we see during the rest of each day, and by missing them there should be no large effect on the measured statistics. Instead, the sequences that are cut off (*incomplete*) because of the gaps could instead skew some statistics, such as sequence duration. We could see in the basic results section (Figure 5.10) that more than 97% of all analyzed sequences were shorter than 15 minutes. For a 15 minute long sequence to be cut of by one of these gaps, it will have to start somewhere in the 15 minutes before the gap, i.e. a period of around 1% of a day [2].

Assuming sequence starting times are uniform during a day, a higher bound is then that for 97% of all sequences, only at most 1% of them are cut off by a gap. Since most sequences are much shorter than 15 minutes, it seems reasonable that less than 1% of all sequences would have been cut off. Longer sequences have a higher risk of being cut off, but then again there are fewer of them. All in all, the gaps should have little effect on the collected statistics and the clustering. However, we could possibly expect that some sequences were cut off in the clusters with long sequence duration, that otherwise would have been longer.

6.1.2 User coverage

We saw in Section 5.1.2 an analysis of how the sets of users seen in different log types overlapped. The combination of log types that yielded the biggest overlapping user set was quite varied between datasets. If you want to undertake studies that correlate different log types for the same set of users, it is important that there is a high enough number of users that appear in both logs. Statistics about these overlaps can indicate which overlaps are big enough to study reliably. If there is a high interest in doing log correlation studies, it could also be helpful to use this kind of statistic to evaluate choices about on which nodes in the network logging is done, since it could be possible to end up with a larger user set overlap between log type recordings, by adjusting the choices of which nodes in the radio network to do logging on.

6.1.3 Data integrity

In Section 5.1.3, we saw some statistics about specific data integrity issues in the radio network event logs. The RNC processing units in the *Asia-A* dataset tended to have a

[1] Possible reasons for this could be missing or corrupt binary raw data files, or some problem in the tools used to process the data before applying the method described in this work

[2] If it would have started during the gap, we would miss the RRC Connection Request event, and discard the whole sequence, and the effect is the same as if a complete sequence was missed during a gap

higher error rate than in the other datasets. At the same time we could observe that the cause for such non-logging time in this dataset to a large extent was caused by file size limits being overrun, an error trigger that was not seen in the other datasets. It is possible that with the file issues aside, the error rates would have been comparable to the other datasets studied.

On the whole, however, the effect of these issues should be quite low, since we saw that at least 97% of the processing units were without issues around 99 % of the time.

6.2 Comparing the different datasets

From the basic statistics results presented in Section 5.2 we could note a number of clear differences between the four datasets.

The *Asia-B* dataset is extreme in many respects. It has substantially much more radio network event sequences with long durations than the other datasets, and also more sequences with a high number of events. Looking at specific event types we can see that the distributions for *Asia-B* are especially ahead in HS-DSCH cell changes and channel switches, fairly ahead in cell updates and not that much ahead in soft handovers. We could also note that the differences were mainly in the upper quartile, while median value for *North America* was similar and often slightly higher than for *Asia-B*.

Another statistic where *Asia-B* really differentiated itself from the other datasets was in the traffic volume per user, with a median 50 times higher than any other dataset. One can imagine different causes for this difference, such as user behavior, the kind of area covered by the recording (urban/rural) or the behavior of the common applications used in this area. In this thesis we will simply conclude that there is a significant difference, which could be of value for someone who aims to further analyze these datasets.

We could see that more than 75% of users in *Asia-B* were using Android/iOS based devices, while the corresponding numbers for the other datasets were between 25 and 50%. The sophistication of these two operating systems as well as the large number of apps available might explain the extreme statistics recorded for *Asia-B* to an extent. If the users have more useful devices they might use them more and longer each time.

As mentioned, for *North America* we could note that the median was comparable or higher to *Asia-B* for radio network sequence duration, length and number of channel switch and cell updates per sequence. While having reasonably high median values, the sequence duration for in the *North America* dataset was actually lower than for *Asia-A* and *EU* at the 90th percentile.

Asia-A and *EU* appeared very similarly distributed for most statistics, and usually had a lower distribution of values than *NA* and considerably lower than *Asia-B*. Notable differences between *Asia-A* and *EU* are that *Asia-A* had twice as many sequences with HS-DSCH cell changes than *EU*, and a higher median traffic volume per user.

Looking at the device fleets, we see that PCs had a much higher portion of the user devices in *Asia-A* and *EU* (around 10%) than in the other datasets (less than 2%). The PC category also managed to account for around 75% of the traffic volume in *Asia-A* and *EU*, which instead was dominated by handheld devices in *North America* and *Asia-B*.

We could also note that *EU* had only a small fraction of iOS/Android devices (around 20%) and instead a much higher portion of devices with proprietary OSs (feature phones) and Symbian based devices. These generally do not have as much functionality as Android/iOS based devices. A point to consider is also that the *Asia-A* recording is from 2010, so that devices listed with iOS are to a larger extent older IPhone models than those in the other datasets. Similarly, with *Asia-B* being the most recent recording, from 2013, an iOS and Android device OS label might imply newer phone models than in the other datasets.

6.3 Cluster properties

The clusters found in the analysis of common radio network sequences all had distinguishing properties. Most were distinguishable by the counts of the four events selected as features, while some had similar counts but different sequence duration (the fifth raw feature). We could see that the clusters with the lowest median sequence duration together contained most of the sequences.

Some clusters were more unique than others. In Cluster 9 e.g., the median duration was quite long, but compared to other clusters with a long median duration (clusters 8 and 10), the number of events per sequence was significantly lower. Another aspect differentiating Cluster 9 from 8 and 10 was that its sequences contained very few soft handovers and HS-DSCH cell changes, but instead mostly just channel switches and cell updates.

Another fairly unique cluster was number 7, whose sequences contained a high number of soft handover events compared to both Cluster 6 and 8 which had similar median duration.

On the opposite side of the spectrum there were two clusters that were very similar in some respects: clusters 5 and 6. They had an almost identical distribution of sequence duration, number of channel switches and number of cell updates, but differed in that cluster 5 had very few soft handovers and HS-DSCH cell changes.

6.4 Cluster distributions over datasets

As we could see in Figure 5.35, there were few clusters where there was a similar proportion of each dataset. Only in Cluster 1 and 2 were there similar proportions. On the other hand, considering the cluster sizes presented in Table 5.9, these clusters together cover 35% of all sequences. In clusters 0, 3 and 4 the proportions are not substantially skewed, but we cannot say that they are similar either. Using four datasets might not give us enough evidence to clearly show the existence of universally occurring patterns in radio network sequences, but from the data collected in this report, it seems Cluster 1 and 2 could be candidates for universal patterns.

Cluster 1 can be characterized as having sequences with a duration around 1.5 seconds, with three events, typically in the sequence: RRC connection request, IMSI registered at RNC and IU Release. Cluster 2 is similar, but its sequences typically also contain a soft handover event, usually just after the RRC connection request, and have a duration around 2 seconds.

If we instead consider clusters with a very skewed distribution over datasets we find several interesting cases. *North America* is dominating both Cluster 5 and 6, with more than half of all sequences in each. We also found a dense range of sequences with 25% of the sequences in the *North America* dataset being around 10-15 s, which fits very well with the duration distributions of clusters 5 and 6. We then have some fairly good pointers on how sequences in this group look, with the typical events sequences being listed in Appendix B, Table B.7 and B.8, or more visually by the example sequences in Figure 5.26.

We also found that clusters 8, 9 and 10 are dominated by *Asia-B*, having around 50%, 75% and 85% respectively of all the sequences in each of these clusters. This is not surprising considering that these are the clusters with the highest median durations, and that *Asia-B* had more sequences with long durations than the other datasets. Looking at the properties of these specific clusters however gives us a more detailed view on how these long sequences look like. While clusters 8 and 10 have a fairly high number of all four key event types, the sequences in Cluster 9 almost exclusively have channel switches and cell updates. It is also interesting to note that the small concentration of sequences around 600 s of duration in *Asia-B* observed in Figure 5.10 could be linked quite strongly to Cluster 9. This is especially important since sequences of this duration also could have fit in quite well in both Cluster 8 and 10, but that we noticed that this specific group of sequences to a large extent ended up in Cluster 9. Going from a small bump in a distribution from basic statistics we can from the analysis of common sequences also find out significant properties of this concentration of sequences, such as what events they mainly contain, which order the events tend to occur in and what devices mainly are responsible for generating them.

6.5 Evaluation of clustering results

While we have found that the clusters had fairly different properties, these properties concerned the cluster centers, and does not say much about if the sequences in a cluster lie mostly close to the center, or if they are more uniformly distributed over the space we are clustering in. In Section 5.3.1 the results of the Gap Statistic test were presented, which compared the results from clustering on the actual data, to data meant to represent unclusterable, uniformly distributed data. The results showed that the highest gap for the error between these two clustering scenarios occurred when using $K = 11$ clusters. However, as the test is designed to only recommend a higher number of clusters to be used if there is substantial evidence, the result of the full Gap Statistic procedure was to recommend $K = 1$ clusters, i.e. that the data should not be clustered.

This might seem like a discouraging result, but we should bear in mind that the test is quite strict being designed, as the authors claim, to find “well-separated clusters” [12]. From the results found we can conclude that there is not strong evidence for the existence of *well-separated* clusters in the data.

From the perspective of trying to compare differences between datasets it is not a strong requirement that there are well-separated clusters in the data. The clustering gives us a way to divide the multidimensional space that we have placed the sequences in into regions. Recall that each point is assigned to the cluster with the nearest cluster center. If we instead of the found cluster centers used 11 random points as cluster centers and used this to divide up the sequences, this would still give us a somewhat meaningful model for comparing similarities and differences of the datasets. The point of using clustering is to try to find a more useful division of the space, by looking for natural groupings of similar sequences. The Gap Statistic has therefore been used as guidance in finding a suitable number of clusters to use (11) which results in somewhat well-separated clusters.

As we could see in the different perspectives presented of the clusters, but perhaps most clearly in the visualization in Figure 5.26, we have found clusters with quite different properties, which indicates that the clustering has been successful at some level. Of course we can also start to reflect on if the distance measure that we ended up with accurately reflects when sequences are similar and different. We should however be aware that there can be several different sets of features that all generate results that are meaningful in their own way. We can find an example of this in [16], which brings up a cluster scenario where two different feature sets generate two different meaningful clusterings. The scenario was to cluster animals by their attributes, and when using a certain set of attributes as features the clustering managed to group mammals separately from birds, while another set of attributes (features) led to seeing the separation between predators and non-predators instead.

In our case we could imagine picking different features or doing other transforms that affect how we value these features. The selection of features here focused on the occurrences of four key events. It is easy to instead study other types of events, and perhaps even split up events into different categories, e.g. a channel switch into an event type for switching to a HS channel and another type for the other channels. Another option would be to use rates of event occurrences per time unit, instead of the counts of events per sequence that was used in this work. This might yield very different results, but care has to be taken if you also use duration and the total number of events per sequence as features, since the features then become correlated in ways that can be hard to reason about.

As a reflection on the distance measure used we can consider clusters 5 and 6. We saw that they were similar in many respects, but cluster 6 had more soft handovers and HS-DSCH cell changes. Depending on your point of view this difference might be important or insignificant, which highlights the possibility to adapt the method to specific purposes by changing the sequence similarity definition.

6.6 Contributions of the used method

The datasets used in this report and other similar datasets are already being studied from different angles at Ericsson. While the basic statistic analysis in this thesis project does not apply any novel techniques for this part, one of the results of the project is a set of reusable scripts that can extract a repeatable set of statistics between datasets of the type studied, producing a report that presents a comparative view of the dataset statistics.

Another valuable part of this project has been to extract dataset *metadata*, that describes the dataset in terms of time span, coverage and data integrity. We earlier concluded that the effect of the gaps and event log errors was not very significant in the studied datasets, which is important when considering the reliability of the statistical analysis that has been done. The user set overlap analysis provides pointers to which kind of log correlation studies can be undertaken with a solid coverage in the data. The dataset metadata can also be valuable as feedback into the data collection loop, to follow up problems with specific recordings.

The key novel contribution of this work is however the analysis of common radio network event sequences through clustering. Through this work we get an idea of what groups of similar sequences exist in the data, across the different datasets. The cluster example sequence plot in Figure 5.26 provides a novel way of visualizing radio network events, which highlights the differences of the found clusters.

Studying the clusters found by the method allows us to explore differences indicated by the basic statistics in more detail. We found e.g. that the *Asia-B* dataset tends to have radio network event sequences of longer duration than the other datasets. As we could expect, we then saw that the clusters with a high median sequence duration were dominated by *Asia-B*. Being able to look at specific properties for these clusters, however, provides a much more detailed view of this difference from the datasets. We can now find what event patterns are common among the sequences with long duration, and also subdivide them into groups with different properties. Cluster 9 e.g. had very few soft handovers and HS-DSCH cell changes compared to clusters 8 and 10.

A slightly unexpected result was the opportunity to link unique phenomena discovered among the basic statistics to specific clusters. We could see that the *North America* dataset had a large concentration of sequences with durations around 10-15 seconds, which we could then link fairly well to clusters 5 and 6. This link allows us to connect the cluster properties to this concentration, such as concluding that its sequences most likely have one of the event patterns listed in Appendix B, Table B.7 and B.8.

The link found between a small concentration of sequences in the *Asia-B* dataset with a duration around 600 seconds, to Cluster 9 is of specific interest. There are three clusters that have a significant amount sequences with this duration, but we could uniquely determine that the concentration found ended up in Cluster 9. As mentioned earlier, Cluster 9 had sequences with a unique event composition compared to clusters 8 and 10, so this concentration of sequences has both a fairly specific event composition and duration.

It is perhaps not surprising that concentrations of some values seen in the basic statistics end up in one cluster. After all, the clustering is supposed to group similar sequences with each other.

Chapter 7

Conclusions

7.1 Key findings

Four datasets were analyzed both in terms of basic statistics and which groupings of common radio network event sequences could be found through clustering.

When comparing the basic statistics of the studied datasets, some clear differences were discovered. The most recent dataset, *Asia-B* from 2013, was found to have radio network event sequences with markedly longer durations than the other datasets. We could also see that the median traffic volume per user was 50 times higher than in any of the other datasets. We can speculate if it could be usage patterns of the users, the behavior of the type of devices used in this network or possibly patterns from how certain popular applications communicate that account for these differences. We could note that the *Asia-B* dataset had a high portion of devices with Android and iOS operating system compared to the other datasets, which could account for some of the differences.

From the basic statistics we could also note that the *Asia-A* (from 2010) and *EU* (from 2012) datasets tended to have similar distributions for most of the measured statistics. Compared to the *Asia-B* dataset they had less extreme values, e.g. having more radio network event sequences with short durations and much smaller traffic volume per user. The *North America* dataset (from 2012) tended to have distributions in between *EU/Asia-A* and *Asia-B*, although for some statistics the distribution was very similar to the one in *Asia-B* up to the median.

The analysis of common radio network event sequences through clustering managed to find clusters with distinguishing features. The Gap Statistic was used as guidance to choose to use $K = 11$ clusters, since the difference in error when clustering on the given data compared to a reference distribution of "unclusterable" data was the highest when using 11 clusters. The specific recommendation of the Gap Statistic method, however, was to use $K = 1$ clusters, which is taken to mean that no strong evidence was found for the existence of well-separated clusters in the data.

The found clusters and their properties proved a useful tool in further exploring differences indicated by the basics statistics. The *Asia-B* dataset was found to dominate the clusters with long sequence durations, which could be expected since the longer durations were indicated by the basic statistics. By studying the properties of these clusters we could further characterize the long sequences that were most common in the *Asia-B* dataset. Here we saw that the long sequences not just were divided up into clusters based on how long they were, but also based on the occurrence count of specific events. Cluster 9 represented long sequences with very few soft handovers and HS-DSCH cell changes.

A specifically interesting link was found between a small concentration of sequences with durations around 600 seconds in *Asia-B* and Cluster 9. The clear majority of these sequences ended up in Cluster 9, which as mentioned was unique in its lack of soft handovers and HS-DSCH cell changes.

The collection of dataset metadata has provided a means to judge the data integrity of a dataset and give pointers to what potential log correlation studies can be undertaken. This kind of data is important for a potential future analyst to decide of a dataset is at a basic level suitable for a particular study.

In conclusion, the work in this thesis project allowed key differences between the studied datasets to be discovered from the basic statistics, and then be further explored by considering how the radio network event sequences from the datasets were distributed over the found clusters and what properties these clusters had.

7.2 Future studies

The work done in this thesis project introduces some new ways to approach the existing datasets, but one can imagine several ways to further develop these ideas.

7.2.1 Dealing with many plots

The plots presented in this report are a subset of the plots generated by the tool developed as a part of the project. Even when just considering the plots that are included in the report it is hard to get a good overview of the properties of the different datasets. While the cumulative distribution function plots for multiple datasets allow for easy comparison on a particular statistic, during the course of this work it was found that you often want to view and link several plots together at the same time.

It would be helpful to have a tool for interactively exploring plots that are linked together. One can imagine an interface where you can interactively hide or show distributions in a plot, such as the radio network event sequence duration for both datasets and clusters.

Another useful piece of functionality would be to have more automatic follow up on features in the basic statistics when looking at cluster properties. You could for example be able to mark a certain range for a statistic, say sequences with duration 10-20 seconds, and then get a distribution for the sequences in this range over the clusters. This would greatly simplify the process of linking features from the basic statistics to clusters.

A different approach to the problem of having to go through many plots to get an idea of the similarities and differences of the studied datasets would be to try to define some standardized criteria that you can judge datasets by, such as user mobility, HS channel usage, smartphone penetration, etc., perhaps by aggregating several statistics.

7.2.2 Possible adjustments to the clustering method

The distance measure used by the clustering method is a result of the choice of the five features, the specific normalization and transformation methods and usage of the Euclidean distance measure. One can easily imagine many other features and possible transforms. Experts trying to use the clustering results to compare datasets based on the dataset sequence distribution over the resulting clusters should perhaps also try to reflect on if the distance measure used fits with their idea of when two sequences are similar and different.

The aim of this thesis project was to use a fairly general similarity measure that was not too complex. If you have a more specific purpose there are good possibilities to tailor the distance metric to another idea of similarity.

One can try to find more information about each sequence that could either be used as a feature or just simply when analyzing cluster properties. You could also skip clustering altogether and instead focus on which variables are correlated and try to speculate in causality. Does e.g. a high smartphone penetration imply long durations for radio network event sequences?

There are also several other different entities that you can try to cluster in these datasets. You could try to cluster users based on their usage patterns (possibly simply by considering their sequences), or define event sequences in other log types, such as the packet header logs.

In this thesis project the K-means clustering algorithm was used, but there are numerous other clustering methods that have other properties. Density based clustering methods such as DBSCAN do not have the same limitations of the cluster geometry as K-means methods [16]. It could be interesting to experiment with such methods to see if similar clusters are found.

In order to get more universally valid results with the method used in this project it would be interesting to try to use more datasets. Then you have more evidence to support claims about global sequence patterns. Having more cases can also make it easier to discover if some abnormality in a dataset is related to another of the studied variables or is a quirk with a particular dataset. In our case, e.g. *North America* was the dataset having most users with Android devices, and it was dominating two clusters of sequences. It is here difficult to determine if features of the sequences in these clusters are caused by Android properties, or something specific to the dataset, such as hardware configuration, network node geography or user patterns.

Bibliography

[1] Harri Holma and Antti. Toskala. *WCDMA for UMTS.* Wiley, 2000.

[2] Pekka HJ Perala, Antonio Barbuzzi, Gennaro Boggia, and Kostas Pentikousis. Theory and practice of RRC state transitions in UMTS networks. In *GLOBECOM Workshops, 2009 IEEE*, pages 1–6. IEEE, 2009.

[3] Harri Holma and Antti Toskala. *HSDPA/HSUPA for UMTS: High speed radio access for mobile communications.* Wiley. com, 2007.

[4] 3GPP. Numbering, addressing and identification. TS 23.003, 3rd Generation Partnership Project (3GPP), September 2008.

[5] D Knuth. The art of computer programming, volume 2: seminumerical algorithms. *AMC*, 10:12, 1997.

[6] BP Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962.

[7] Yufei Tao. Cmsc 5705 advanced topics in database systems, lecture 7: Reservoir sampling. `http://www.cse.cuhk.edu.hk/~taoyf/course/5705f10/lec7.pdf`, February 2014.

[8] Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The planar k-means problem is NP-hard. In *WALCOM: Algorithms and Computation*, pages 274–285. Springer, 2009.

[9] Stuart P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–136, 1982.

[10] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[11] Suresh Venkatasubramanian. Choosing the number of clusters I: The elbow method. `http://geomblog.blogspot.com/2010/03/this-is-part-of-occasional-series-of.html`, July 2013.

[12] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.

[13] David J Ketchen and Christopher L Shook. The application of cluster analysis in strategic management research: an analysis and critique. *Strategic management journal*, 17(6):441–458, 1996.

[14] Selim Aksoy and Robert M Haralick. Feature normalization and likelihood-based similarity measures for image retrieval. *Pattern Recognition Letters*, 22(5):563–582, 2001.

[15] Elke Achtert, Sascha Goldhofer, H-P Kriegel, Erich Schubert, and Arthur Zimek. Evaluation of clusterings–metrics and visual support. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pages 1285–1288. IEEE, 2012.

[16] Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.

Appendix A

User sampling

Distribution comparision for different user sampling fractions on the NA1 dataset.

Table A.1: Comparison of percentiles for sequence length at different user fractions

	Distribution percentiles								
User fraction	10%	20%	30%	40%	50%	60%	70%	80%	90%
100 %	3.000	3.000	4.000	4.000	4.000	5.000	6.000	7.000	9.000
10 %	3.000	3.000	4.000	4.000	4.000	5.000	6.000	7.000	9.000
1 %	3.000	3.000	4.000	4.000	4.000	5.000	6.000	7.000	9.000

Table A.2: Comparison of percentiles for sequence duration at different user fractions

	Distribution percentiles								
User fraction	10%	20%	30%	40%	50%	60%	70%	80%	90%
100 %	0.830	1.410	1.862	2.614	5.486	7.973	9.692	19.994	28.180
10 %	0.833	1.420	1.885	2.632	5.593	7.994	9.829	20.495	28.136
1 %	0.811	1.409	1.867	2.605	5.525	8.000	9.865	20.291	28.370

Table A.3: Comparison of percentiles for number of channel switches per sequence at different user fractions

	Distribution percentiles								
User fraction	10%	20%	30%	40%	50%	60%	70%	80%	90%
100 %	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	2.000
10 %	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	2.000
1 %	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	2.000

Table A.4: Comparison of percentiles for number of soft handovers per sequence at different user fractions

	Distribution percentiles								
User fraction	10%	20%	30%	40%	50%	60%	70%	80%	90%
100 %	0.000	0.000	0.000	0.000	0.000	0.000	1.000	1.000	2.000
10 %	0.000	0.000	0.000	0.000	0.000	0.000	1.000	1.000	2.000
1 %	0.000	0.000	0.000	0.000	0.000	0.000	1.000	1.000	2.000

Table A.5: Comparison of percentiles for number of cell updates per sequence at different user fractions

	Distribution percentiles								
User fraction	10%	20%	30%	40%	50%	60%	70%	80%	90%
100 %	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000
10 %	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000
1 %	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000

Table A.6: Comparison of percentiles for number of HS-DSCH cell changes per sequence at different user fractions

	Distribution percentiles								
User fraction	10%	20%	30%	40%	50%	60%	70%	80%	90%
100 %	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
10 %	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1 %	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Appendix B

Top event patterns for clusters

The short labels used to identify event types in this section are presented in Table B.1. The top three event patterns in clusters 0-10 are presented in Tables B.2 - B.12.

Table B.1: Short labels for event type

Event type	Short label
RRC connection request	RRC
IMSI registered at RNC	REG
IU Release	IUR
Soft handover	SHO
Channel Switch	CSW
HS-DSCH cell change	HCC
RAB establishment	RAB
Cell update	CUP

Table B.2: Top sequences in Cluster 0

Rank	Fraction	Count	Events
1	1.000	85298	[RRC]
2	0.000	1	[RRC, HCC]
Total	1.000	85299	

Table B.3: Top 3 sequences in Cluster 1

Rank	Fraction	Count	Events
1	0.460	383097	[RRC, REG, IUR]
2	0.153	127019	[RRC, IUR, REG]
3	0.083	69102	[RRC, REG, REG, IUR, IUR]
Total	0.696	579218	

Table B.4: Top 3 sequences in Cluster 2

Rank	Fraction	Count	Events
1	0.187	76426	[RRC, SHO, REG, IUR]
2	0.104	42664	[RRC, REG, SHO, IUR]
3	0.078	31862	[RRC, REG, REG, IUR, SHO, IUR]
Total	0.369	150952	

Table B.5: Top 3 sequences in Cluster 3

Rank	Fraction	Count	Events
1	0.055	13844	[RRC, REG, RAB, SHO, SHO, IUR]
2	0.050	12534	[RRC, SHO, REG, RAB, SHO, IUR]
3	0.048	12151	[RRC, SHO, SHO, REG, IUR]
Total	0.153	38529	

Table B.6: Top 3 sequences in Cluster 4

Rank	Fraction	Count	Events
1	0.791	607667	[RRC, REG, RAB, IUR]
2	0.105	80864	[RRC, REG, RAB, CSW, IUR]
3	0.026	20254	[RRC, RAB, IUR]
Total	0.922	708785	

Table B.7: Top 3 sequences in Cluster 5

Rank	Fraction	Count	Events
1	0.309	102359	[RRC, REG, RAB, SHO, IUR]
2	0.136	45052	[RRC, SHO, REG, RAB, IUR]
3	0.086	28317	[RRC, REG, RAB, SHO, SHO, IUR]
Total	0.531	175728	

Table B.8: Top 3 sequences in Cluster 6

Rank	Fraction	Count	Events
1	0.385	316535	[RRC, REG, RAB, CUP, CSW, IUR, CSW]
2	0.174	143000	[RRC, REG, RAB, CUP, CSW, IUR]
3	0.079	65087	[RRC, REG, RAB, SHO, CUP, CSW, IUR, CSW]
Total	0.638	524622	

Table B.9: Top 3 sequences in Cluster 7

Rank	Fraction	Count	Events
1	0.086	20133	[RRC, REG, RAB, SHO, SHO, CUP, CSW, IUR, CSW]
2	0.027	6368	[RRC, SHO, REG, RAB, SHO, CUP, CSW, IUR, CSW]
3	0.025	5901	[RRC, REG, RAB, SHO, SHO, CUP, CSW, IUR]
Total	0.138	32402	

Table B.10: Top 3 sequences in Cluster 8

Rank	Fraction	Count	Events
1	0.040	3751	[RRC, REG, RAB, SHO, SHO, SHO, SHO, IUR]
2	0.028	2658	[RRC, REG, RAB, SHO, SHO, SHO, SHO, SHO, SHO, IUR]
3	0.025	2293	[RRC, REG, RAB, SHO, SHO, SHO, SHO, SHO, IUR]
Total	0.093	8702	

Table B.11: Top 3 sequences in Cluster 9

Rank	Fraction	Count	Events
1	0.109	13242	[RRC, REG, RAB, CUP, CSW, CSW, IUR, CUP, CSW]
2	0.045	5466	[RRC, REG, RAB, CUP, CSW, CSW, CUP, CSW, CSW, CUP, CSW, IUR, CSW]
3	0.024	2940	[RRC, REG, RAB, CUP, CSW, CSW, CUP, CSW, CSW, IUR, CUP, CSW]
Total	0.178	21648	

Table B.12: Top 3 sequences in Cluster 10

Rank	Fraction	Count	Events
1	0.013	646	[RRC, REG, RAB, CUP, CSW, CSW, CUP, CSW, CSW, CUP, CSW, CSW, ...]
2	0.001	49	[RRC, REG, RAB, SHO, SHO, CUP, CSW, CSW, CUP, CSW, CSW, CUP, ...]
3	0.001	41	[RRC, REG, RAB, SHO, CUP, CSW, CSW, CUP, CSW, CSW, CUP, CSW, ...]
Total	0.015	736	

Appendix C

Additional extracted metrics

This is an extension of the list in Section 4.4.2. These are the additional metrics that are also collected by the extraction tool but were not addressed in this report.

The metrics marked with † are extracted as category counts, while the others are extracted as distributions as described earlier in this section. Wherever *EVENT-TYPE* is mentioned below it indicates that the metric is recorded separately for each key event listed below. The events were chosen to both cover changes to the communication state and mobility updates (see Section 2.1.2-2.1.3):

- Channel switch
- Soft handover
- Cell update
- HS-DSCH cell change

Similarly, metrics with *CHANNEL-TYPE* are repeated for the following downlink channels:

- 16 kbps
- 64 kbps
- 128 kbps
- 384 kbps
- HS-DSCH

In the same manner we also have *TRAFFIC-DIRECTION* being replaced by, respectively:

- Uplink
- Downlink
- Total (Uplink + Downlink) [1]

The additional metrics by log type:

- Event logs
  - ***EVENT-TYPE* event frequency**: Counts key events per minute within radio network event sequences.

[1] The total for some metrics was also already included in Section 4.4.2

  - **Duration in *CHANNEL-TYPE* downlink channel per sequence**: Captures how much different downlink channel types are used in each sequence.
  - **User connection frequency**: This is measured as number of started radio network event sequence per user per 24 h, and can indicated any imbalances, e.g. that a small number of users is responsible for most data connections.
  - **Frequency of *EVENT-TYPE* per user**: This is measured as events of the given type for a user per minute during the active time for that user, i.e. the time the user has an active data connection.

- Summary logs
  - **Summary activity traffic volume in *TRAFFIC-DIRECTION***: Measures the number of bytes communicated during a summary activity. This can indicate different usage patterns that might be caused by user behavior or the kind of application being used.
  - **Traffic volume per user in *TRAFFIC-DIRECTION***: The total amount of bytes sent or received by a user during the measurement period.
  - **Functionality by traffic volume in *TRAFFIC-DIRECTION*** (†): A catergory count of the total amount of bytes sent or received added up for different device OS's. Could be e.g. *Android*, *iOS*, *Symbian* or *Blackberry*.
  - **Device types by traffic volume in *TRAFFIC-DIRECTION*** (†): A category count of the total amount of bytes sent or received added up for different device types. Device type is one of *HANDHELD*, *M2M*, *PC*, *ROUTER* or *TABLET*.
  - **Device OS by traffic volume in *TRAFFIC-DIRECTION*** (†): A category count of the total amount of bytes sent or received added up for different device OS's. Could be e.g. *Android*, *iOS*, *Symbian* or *Blackberry*.
  - **Social networking providers by traffic volume in *TRAFFIC-DIRECTION*** (†): A catergory count of the total amount of bytes sent or received in activities where the functionality is *social-networking*, added up for different service providers. Could be e.g. *Facebook* or *Twitter*.
  - **Video providers by traffic volume in *TRAFFIC-DIRECTION*** (†): A catergory count of the total amount of bytes sent or received in activities where the functionality is *video*, added up for different service providers. Could be e.g. *YouTube* or *Netflix*.
  - **Protocol by traffic volume in *TRAFFIC-DIRECTION*** (†): A category count of the total amount of bytes sent or received added up for different protocols. Could be e.g. *HTTP*, *BitTorrent* or *POP3*.

- Flow logs
  - **Flow data volume in *TRAFFIC-DIRECTION***: The number of bytes sent in the given direction during a flow.
  - **Traffic volume per user in *TRAFFIC-DIRECTION***: The total amount of bytes sent or received by a user during the measurement period.
  - **Functionality by traffic volume in *TRAFFIC-DIRECTION*** (†): A catergory count of the total amount of bytes sent or received added up for different device OS's. Could be e.g. *Android*, *iOS*, *Symbian* or *Blackberry*.

  - **Device types by traffic volume in *TRAFFIC-DIRECTION* (†)**: A category count of the total amount of bytes sent or received added up for different device types. Device type is one of *HANDHELD*, *M2M*, *PC*, *ROUTER* or *TABLET*.
  - **Device OS by traffic volume in *TRAFFIC-DIRECTION* (†)**: A category count of the total amount of bytes sent or received added up for different device OS's. Could be e.g. *Android*, *iOS*, *Symbian* or *Blackberry*.
  - **Social networking providers by traffic volume in *TRAFFIC-DIRECTION* (†)**: A catergory count of the total amount of bytes sent or received in activities where the functionality is *social-networking*, added up for different service providers. Could be e.g. *Facebook* or *Twitter*.
  - **Video providers by traffic volume in *TRAFFIC-DIRECTION* (†)**: A catergory count of the total amount of bytes sent or received in activities where the functionality is *video*, added up for different service providers. Could be e.g. *YouTube* or *Netflix*.
  - **Protocol by traffic volume in *TRAFFIC-DIRECTION* (†)**: A category count of the total amount of bytes sent or received added up for different protocols. Could be e.g. *HTTP*, *BitTorrent* or *POP3*.

- Packet header logs
  - **Traffic volume per user**: The total amount of bytes sent or received by a user during the measurement period.
  - **Device types by traffic volume (†)**: A category count of the total amount of bytes sent or received added up for different device types. Device type is one of *HANDHELD*, *M2M*, *PC*, *ROUTER* or *TABLET*.
  - **Device OS by traffic volume (†)**: A category count of the total amount of bytes sent or received added up for different device OS's. Could be e.g. *Android*, *iOS*, *Symbian* or *Blackberry*.
  - **Social networking providers by traffic volume (†)**: A catergory count of the total amount of bytes sent or received in activities where the functionality is *social-networking*, added up for different service providers. Could be e.g. *Facebook* or *Twitter*.
  - **Video providers by traffic volume (†)**: A catergory count of the total amount of bytes sent or received in activities where the functionality is *video*, added up for different service providers. Could be e.g. *YouTube* or *Netflix*.
  - **Protocol by traffic volume (†)**: A category count of the total amount of bytes sent or received added up for different protocols. Could be e.g. *HTTP*, *BitTorrent* or *POP3*.

Appendix D

User set overlap details

This appendix provides more detailed results related to the overlaps between the sets of users observed in the different log types of each dataset. See Section 5.1.2.

Tables D.1-D.4 present information about the size of the user sets, and the pairwise overlap between log types. In these tables, for rows with two log types listed, the *Size* column refers to the number of users appearing in both of these log types in the recording.

Table D.1: Overview of user set sizes and pair-wise overlaps between different log types (Asia-A)

Log types	Size	All %	Event %	Flow %	Summary %	PDP %
All	1269428	100%	-	-	-	-
Event	1233505	97.17%	100.00%	-	-	-
Flow	254869	20.08%	-	100.00%	-	-
Summary	130577	10.29%	-	-	100.00%	-
PDP	254911	20.08%	-	-	-	100.00%
Event, Flow	218988	17.25%	17.75%	85.92%	-	-
Event, Summary	110398	8.70%	8.95%	-	84.55%	-
Event, PDP	218988	17.25%	17.75%	-	-	85.91%
Flow, Summary	130575	10.29%	-	51.23%	100.00%	-
Flow, PDP	254869	20.08%	-	100.00%	-	99.98%
Summary, PDP	130575	10.29%	-	-	100.00%	51.22%

Table D.2: Overview of user set sizes and pair-wise overlaps between different logtypes (EU)

Logtypes	Size	All %	Event %	Flow %	Summary %	PDP %	Packet %
All	838457	100%	-	-	-	-	-
Event	447092	53.32%	100.00%	-	-	-	-
Flow	524406	62.54%	-	100.00%	-	-	-
Summary	525237	62.64%	-	-	100.00%	-	-
PDP	658732	78.56%	-	-	-	100.00%	-
Packet	137159	16.36%	-	-	-	-	100.00%
Event, Flow	231092	27.56%	51.69%	44.07%	-	-	-
Event, Summary	231301	27.59%	51.73%	-	44.04%	-	-
Event, PDP	269359	32.13%	60.25%	-	-	40.89%	-
Event, Packet	80644	9.62%	18.04%	-	-	-	58.80%
Flow, Summary	524406	62.54%	-	100.00%	99.84%	-	-
Flow, PDP	521604	62.21%	-	99.47%	-	79.18%	-
Flow, Packet	137157	16.36%	-	26.15%	-	-	100.00%
Summary, PDP	522214	62.28%	-	-	99.42%	79.28%	-
Summary, Packet	137158	16.36%	-	-	26.11%	-	100.00%
PDP, Packet	137024	16.34%	-	-	-	20.80%	99.90%

Table D.3: Overview of user set sizes and pair-wise overlaps between different logtypes (North America)

Logtypes	Size	All %	Event %	Flow %	Summary %	PDP %	Packet %
All	1925920	100%	-	-	-	-	-
Event	376770	19.56%	100.00%	-	-	-	-
Flow	192623	10.00%	-	100.00%	-	-	-
Summary	192618	10.00%	-	-	100.00%	-	-
PDP	1866701	96.93%	-	-	-	100.00%	-
Packet	80981	4.20%	-	-	-	-	100.00%
Event, Flow	35629	1.85%	9.46%	18.50%	-	-	-
Event, Summary	35633	1.85%	9.46%	-	18.50%	-	-
Event, PDP	318969	16.56%	84.66%	-	-	17.09%	-
Event, Packet	18608	0.97%	4.94%	-	-	-	22.98%
Flow, Summary	192603	10.00%	-	99.99%	99.99%	-	-
Flow, PDP	190953	9.91%	-	99.13%	-	10.23%	-
Flow, Packet	80611	4.19%	-	41.85%	-	-	99.54%
Summary, PDP	190948	9.91%	-	-	99.13%	10.23%	-
Summary, Packet	80611	4.19%	-	-	41.85%	-	99.54%
PDP, Packet	79754	4.14%	-	-	-	4.27%	98.48%

Table D.4: Overview of user set sizes and pair-wise overlaps between different logtypes (Asia-B)

Logtypes	Size	All %	Event %	Summary %	PDP %	Packet %
All	1331244	100%	-	-	-	-
Event	862491	64.79%	100.00%	-	-	-
Summary	255928	19.22%	-	100.00%	-	-
PDP	1198738	90.05%	-	-	100.00%	-
Packet	157161	11.81%	-	-	-	100.00%
Event, Summary	180779	13.58%	20.96%	70.64%	-	-
Event, PDP	729986	54.83%	84.64%	-	60.90%	-
Event, Packet	117078	8.79%	13.57%	-	-	74.50%
Summary, PDP	255927	19.22%	-	100.00%	21.35%	-
Summary, Packet	157161	11.81%	-	61.41%	-	100.00%
PDP, Packet	157161	11.81%	-	-	13.11%	100.00%