Efficient proximity detection among mobile clients using the GSM network

Thesis for Master Embedded Systems, departement Computer Science
Design and Analysis of Communication Systems (DACS),
Faculty of Electrical Engineering, Mathematics and Computer Science,
University of Twente

by Ing M.T. Witteman
October 18, 2007

Supervising committee:

Dr. Maarten Wegdam (Alcatel-Lucent, Bell-Labs Europe)
Dr. Ir. Geert Heijenk (University of Twente)
Fei Liu, M.Sc. (University of Twente)
Ir. Jacco Brok (Alcatel-Lucent, Bell-Labs Europe)
Ir. Ing. Harold Teunissen (Alcatel-Lucent, Bell-Labs Europe)

Abstract

Personalized services, such as customized ringtone recommendations and location based services increase the popularity of mobile phones. A location based service (LBS) is a type of personalized service that uses the location of the mobile user. Proximity detection is defined as the capability of a location based service to automatically detect when a pair of mobile users is closer to each other than a certain proximity distance. To realize this function the location of the mobile users have to be permanently tracked.

The main objective of this research is to find to what extent it is possible to use GSM cell-ids for proximity detections among mobile entities. We studied the typical GSM characteristics and the state of the art proximity detection algorithms found in literature. All the proximity algorithms we found use X, Y coordinates as location identifier. Dynamic Circles by George True et. al. was stated in literature to perform the best. This algorithm is used as inspiration for four new algorithms that use the GSM network to implement proximity detection. The proposed algorithms are divided into two groups, the basic group and the graph group. The basic group contains two algorithms that use the information available in the GSM network, called cell-id algorithm and LAC algorithm. The graph group uses a cell-id graph representing all the cell switches, to get more efficient algorithms. This group contains two algorithms, named Graph1 algorithm and Graph2 algorithm.

For the comparison of the algorithms a simulator was developed. The simulator can simulate the amount of messages, the traffic over TCP/IP, missed proximities and the accuracy of proximity distance using GSM. The input data of the simulator are user movement paths. These paths can be generated or can be real logged data gathered by a mobile device.

For a realistic cell topology it was found that the median of differences between real distance and cell-id distance is around 1100 meters. Using GSM cells as location identifier introduces missed proximities, i.e. false negatives. We found that for a realistic cell layer and a proximity distance of 1100 meters the number of missed proximities is 3.5% of the total proximity detections.

A simulation of a distributed scenario, representing rush hour, stated that for low density of users the Graph algorithms outperform the Basic algorithms in terms of generated messages and traffic. From this simulation it is found that for 30 buddies a user consumes a total of 30MB of data traffic per month.

Preface

In front of you is my thesis for concluding my Master Embedded Systems. I preferred to complete the last part of my academic study not at the university. An assignment about context aware application was available at Alcatel Lucent(ALU) Bell-Labs Europe. The assignment was to develop an application that uses context, e.g. calendar, location and temperature, to make life easier. Bell-Labs Europe was interested in the deployment of such an application: should it be a client based application or could it better be more server oriented, or maybe hybrid? Unfortunately, for most of the context aware applications it was already obvious which parts of the application should be client side and which server side. There was no challenge in the assignment.

Jacco Brok (ALU) came with another problem related to context aware applications: The possibility to detect if two mobile users are within the vicinity of each other. After some literature study it was found that people already had done research on so called proximity detection algorithms, to detect if two mobile targets approach each other within a certain predefined distance. The algorithms were efficient, in terms of number of messages, but used GPS as a location identifier on the client side. From the experience of colleagues and from literature we concluded that GPS was not suitable for the application. The application had to run continuously, GPS drains the battery and has poor coverage, e.g. you need to have 'line of sight', inside buildings you will not get a GPS fix or coordinate. Because the application is deployed on mobile GSM devices, the question arose: 'Why not use the GSM network?' The final assignment was born.

I became part of WorkPackage 2: 'context aware applications'. This work package is part of the Awareness project within Freeband. I enjoyed the work meetings; it gave me insight in a large interesting research project with several partners. Working at Alcatel-Lucent, Bell-Labs Europe was very interesting, I enjoyed working at the location in Enschede.

Maarten Witteman
October 2007 Twente

Acknowledgments

A lot of people were involved in my graduation project. They helped me with problems or just gave feedback to the several technical sessions about my work. I especially would like to thank the following persons:

Maarten Wegdam, from Alcatel-Lucent, Bell-Labs Europe, supervised my graduation project, first as member of my committee and later as first supervisor. Maarten has helped me during the second half of the period to finish the project with this thesis. I would like to thank Maarten for his help and time he always took to discuss my problems.

I would also like to thank Jacco Brok for being my first supervisor before he left Alcatel-Lucent. Jacco helped me to form a challenging and interesting assignment and I enjoyed the discussions and useful tips from Jacco during the first half of the assignment.

Geert Heijenk and Fei Lui have supervised me from the university on behalf of the DACS chair. I would like to thank them for their time, helpful tips, discussions and input during the committee meetings and for reviewing my thesis.

Harold Teunissen, former manager of the Bell-Labs department in Enschede, became a member of the committee when Jacco left ALU and Maarten was promoted to first supervisor on behalf of ALU. I would like to thank Harold for his input and support during my graduation period and managing during the whole period. Harold unfortunately, for me, also switched job and left the committee 2 months before I finished my master Thesis.

Erik Meeuwissen and Willem Romijn from the Bell-Labs location in Hilversum helped me understanding the difficulties when developing applications using the GSM network. I would like to thank them for the interesting discussions, feedback and help during the project.

I had a lot of fun with the colleagues at the Bell-Labs location in Enschede and a lot of interesting discussions about my project or other projects, or just innovative ideas. I would like to thank all the colleagues of Bell-Labs Enschede: Ronald, Arjen, Dennis, Dirk-Jaap, Richa and Sue for the relaxing conversations.

I presented a lot of my work in WorkPackage 2 of the Awareness project,

part of Freeband. The people from WP2 gave me useful feedback on the work and problems I presented. I would like to thank them and especially Martin Wibbels and Johan Koolwaaij for providing me access to the Cell-locator database of Telematics Institute Enschede.

I had a lot of fun and relaxing times with my flatmates and study colleagues during my study and graduation. It helped me to relax during my study.

Finally I would like to thank my parents, sister and brother for giving me all the chances I had, for supporting me, for believing in my capabilities and for never doubting me.

Contents

Acronyms

LBS Location Based Service LBS is a service that uses the location of the entity.

GSM Global System for Mobile communication. GSM is the new standard for digital cellular communication in Europe.

GPS Global Positioning System. GPS is the Global Navigation Satellite System. Utilizing a constellation of at least 24 medium Earth orbit satellites that transmit precise microwave signals, the system enables a GPS receiver to determine its location, speed and direction.

cell-id Cell Identifier. cell-id is the cell identifier of an antenna, the mobile phone has this identification number of the antenna it is connected to.

MS Mobile Station. MS is a mobile device that is capable of connecting to a GSM radio cell.

BTS Base Transceiver Station BTS comprises all radio equipment, i.e. antennas, signal processing, amplifiers necessary for radio transmissions

LA Location Area LA all the Base Transceiver Stations connected to the same Mobile Services Switching Center are in the same location area. Each Mobile Services Switching Center forms its own Location Area with its unique identifier called the Location Area Code

LAC Location Area Code LAC unique identifier for Location Areas. A set of neighbouring cell-identifiers form a LAC

BSC Base Station Controller BSC manages the Base Transceiver Stations and handles the handovers from one Base Transceiver Station to another Base Transceiver Station.

XML eXtensible Markup Language XML A general-purpose markup language. It is an extensible language because it allows its users to define

their own tags. Its primary purpose is to facilitate the sharing of data across different information systems, particularly via the Internet

MSC Mobile Services Switching Center. MSC is the digital switch that set up connections to other Mobile Services Switching Centers and to the Base Station ControllerS to provide communication between Mobile Stations

ARPU Average Revenue Per User ARPU This states how much money the company makes from the average user. This is the revenue from the services provided divided by the number of users buying those services

GPRS General Packet Radio Service GPRS Medium to access the internet via an air-interface with transfer rates up to 115 kbit/s. Operators can charge on the ammount of data instead of the connection time.

UMTS Universal Mobile Telecommunications Service UMTS Third Generation (3G) broadband packet-based transmission of text, digitized voice, video and multimedia at data rates up to 2Mbit/s

TCP/IP Transmission Control Protocol / Internet Protocol TCP/IP Offers the functionality for hosts to create connections to other hosts, over which they can exchange streams of data using Stream sockets.

POI Point Of Interests POI is a specific point location that someone may find useful or interesting.

Chapter 1

Introduction

1.1 Motivation

Mobile phones have become part of our lives. We can stay connected wherever we are and whenever we need to with our mobile phones. Originally mobile phones were only used for telephony, nowadays they have evolved to all type of mobile internet services. By personalising mobile services the Average Revenue Per User (ARPU) increases and the churn reduces for an operator. Personalized services can vary from customized ringtone recommendations to Location Based Service (LBS) [HK03]. This thesis focuses on the latter.

Imagine that you are visiting a foreign city and it is lunch time, you can simply use your mobile phone to locate the nearby restaurant [Stea]. Another example of a LBS are the projects E911 and E112. E911 and E112 are both emergency services that will enable mobile, or cellular, phones to process 911 or 112 emergency calls and enable emergency services to locate the geographic position of the caller [UE].

Most of the current LBSs are based on the distance between a mobile entity and a static entity, e.q. Point Of Interests (POI), or just about the location of the mobile user, like the E911 and E112 services. Some LBS services however deal with the distance between two mobile entities, e.g. a dating service that checks your personal interests with other mobile users in your vicinity and "breaks the ice" for you. The focus of this thesis is on a specific subcategory of these LBS services, namely services that notify users when specific other users are in the vicinity.

A typical example is that you are notified if a friend of you is in the vicinity, perhaps you will call your friend to make an appointment. This type of LBS is called proximity detection. This is a typical type of service that runs continuously. Existing implementations of this type of service in literature and/or deployed are all based on Global Positioning System (GPS)

to determine the location of the mobile user. It is very energy consuming to continuously use GPS to determine the mobile users location [Cha98]. On top of that, most mobile phones, on the market, do not have an onboard GPS device.

We therefore focus on an alternative way to determine the users' position using the cell structure/topology of the Global System for Mobile communication (GSM) network as a location identifier. A mobile phone is connected to a GSM network with antennas. Each antenna has a unique number called a cell-id. The mobile phone knows this cell-id and is triggered when the cell-id changes, i.e. switching to another antenna.

1.2 Research Question

Given this introduction to proximity detection and the GSM network a main research question can be formulated:

To what extent is it possible to use GSM cell-ids for proximity detections among mobile entities?

In order to answer this question, and to narrow down the field of research, a number of sub questions are formulated. These sub questions guide the research process and will be answered in the conclusion in section 6.1.

1. What are the characteristics of GSM networks that are relevant for determining the location of users, e.g. cell-id topology and size of cells?

2. What is the impact of oscillation (switching between cell-ids while the mobile client is at one place) on the proximity detection?

3. What are the current implementations of proximity detection algorithms, and are they suitable when GSM cell-ids are used as location identifier?

4. What are efficient proximity detection algorithms using the characteristics of GSM networks?

5. Which set of mobility scenarios is a good representation of the movement of people?

6. What is the difference in accuracy between the proximity distance determined using cell-ids and using GPS?

7. What is the number of missed proximities or false negatives if a GSM network topology is used to detect proximity?

8. What proximity detection algorithms generate the least amount of bandwidth, and how much is this?

1.3 Scope

This section describes the decisions made to scope the research project.

To find the usefulness of cell-ids as location identifier for proximity detection the application 'buddy detection' is used to focus the research. Each user has a list of friends with mobile phones (e.g. the phone book uses tags for friends the user wants to detect). This list of friends is used to form a buddy list, this list states from which friends you want to be notified if they are in the vicinity. Proximity must be detected between those users if they approach each other.

To answer the main objective and sub questions simulations are done on four new algorithms using the GSM network to detect proximity. For these simulations, scenarios are defined that cover realistic movement of people. A distributed scenario, representing rush hour traffic in a city, is used to compare the algorithms, in terms of number of messages and data traffic. This scenario is often used in literature to simulate users [AEM+04, TWK06]. Other simulations are done to answer the subquestions about accuracy, missed proximities and data traffic.

Realistic generated user movement or real movement data retrieved from logs of mobile devices is used for the simulations. The cell layer, representing the GSM network topology, used in the simulations can be a generated hexagonal topology or a real topology. The real topology is taken from a cell-id database from Telematics Institute [KW]. This database is filled with GPS coordinates and the cell-id detected at that point. Different network topologies of different operators are stored in this database. For this research one Dutch operator is used from this database.

In one country different operators coexist. Each is with its own network and own network topology. This research uses cell-ids to provide proximity detection so the network topology is of importance. Since each operator uses a different topology it is not possible to detect proximity between two users subscribed to different operators without creating a mapping of the different topologies. In this research all the users are connected to the same network with one topology of cell-ids. From this network only the GSM cells are used and not the Universal Mobile Telecommunications Service (UMTS)

cells. UMTS cells are typical smaller than GSM cells, but the coverage is still less than the coverage of the GSM network.

Privacy is a major issue when developing LBSs, the privacy aspect of proximity detection, is not discussed in this research.

1.4 Approach

To tackle the questions described above the research is divided in four phases:

- First phase: Literature study. The two main topics that are studied are GSM network characteristics and proximity detection algorithms. Different characteristics, e.g. Cell Identifier (cell-id) handovers, oscillation, network topology, mobile phone data are studied. The state of the art proximity detection algorithms or closely related to proximity detection is investigated. Beside these two main topics for a realistic simulations, a study is done on mobility models and tools that generate user movement using mobility models, i.e. CanuMobiSim and VanetMobisim [Steb, HF].

- Second phase: Define new proximity detection algorithms. Use the gathered knowledge of the first phase to define different proximity detection algorithms that use the GSM network as location identifier.

- Third phase: Simulate using generated mobility paths and real logged data, to find an answer to the amount of messages, data traffic, accuracy between cell-ids and GPS, the number of missed proximities and the amount of data traffic. Develop a simulator that has the following tasks:

  - Link the mobility path generators to the simulation environment
  - Use Mathworks Matlab to implement and simulate the algorithms using the input data of the mobility path generators
  - Implement a realistic cell layer to find an answer about the accuracy and missed proximities.
  - Gather data out of the simulations that are used to evaluate the algorithms.

- Fourth phase: Evaluate and conclude. Use real data, in terms of a realistic topology and real logged user-paths to illustrate the algorithms with real data and draw conclusions using the evaluation of the four new proximity detection algorithms.

1.5 Document structure

The remainder of this thesis is structured as follows. Chapter 2 is titled Related Work and describes the typical characteristics of GSM networks, e.g. handovers/cell switches and oscillation, that are relevant for this research. To illustrate the oscillation the results of an experiment are described after the characteristics of GSM networks. The state of the art proximity detection algorithms from literature are also discussed in Chapter 2. The strong and weak points are mentioned and the algorithms are evaluated if the GSM network is used as location identifier. At the end of chapter 2 an experiment is described to illustrate the high energy consumption of GPS.

After the Related Work four new algorithms that detect proximity using the GSM network are introduced in Chapter 3. These algorithms were developed during this research. The strong and weak points of the algorithms are discussed. The simulation environment is outlined in Chapter 4, the simulation process and the generation of the input data is explained. Chapter 5 cover the simulations that were done to find answers to the research question and sub questions. How the confidence level/interval is calculated 'on the fly' in the simulator is discussed. The accuracy of GPS versus cell-id as location identifier is discussed and the missed proximities or false negatives simulation is discussed. The four algorithms are compared against each other, in terms of messages and data traffic, using a distributed scenario, representing rush hour movement. At the end of Chapter 5 an illustration of the amount of data traffic is given and a simulation run with real logged data is given. Chapter 6 concludes the research and presents future work.

Chapter 2

Related work

This chapter describes the related work. First the characteristics of GSM networks are discussed, that are relevant for proximity detection. Next, current proximity detection algorithms we found in literature are described, i.e. Strips, Dynamic circles and Dead-reckoning. These algorithms are based on latitude and longitude coordinates, e.q. GPS, of the users. For each of the proximity detection algorithms, found in literature, the issues there are with applying these algorithms to GSM cell-ids are discussed. At the end this chapter illustrates the energy consumption of GPS on a high-end mobile device.

2.1 GSM network characteristics

This section outlines characteristics of GSM networks that are relevant for proximity detection, e.g. the topology, GSM cell-id switches and oscillation.

2.1.1 Network topology

To explain the network topology some background information is needed about the system architecture of GSM. GSM is a cellular network. A cellular network is a radio network made up of a number of transmitters, called a Base Transceiver Station (BTS), each with its own radio cell. This radio cell can be sectorized resulting in multiple cell-ids per radio cell. The advantages of using more cells instead of one big cell to cover the same area are:

- increased capacity
- reduced power usage
- better coverage

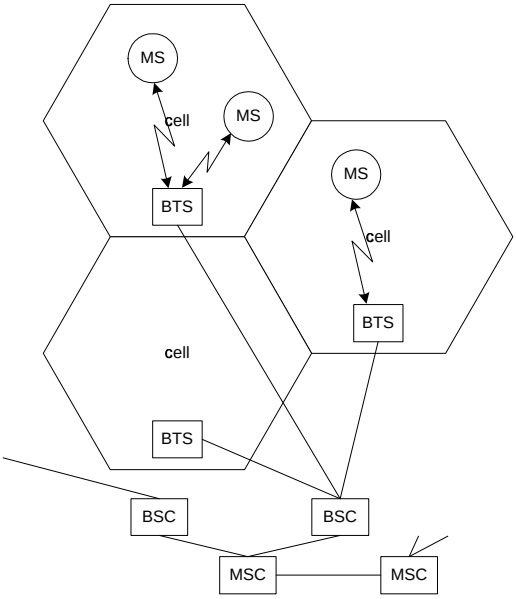


Fig. 2.1: Functional architecture GSM network

A Mobile Station (MS) is connected to a BTS which is connected to a Base Station Controller (BSC) and Mobile Services Switching Center (MSC). The MSC set up connections to other MSCs and to the BSCs to provide communication between two MS registered to different radio cells. Each MS has the responsibility to search for cells in the immediate vicinity. A typical functional architecture of a GSM system is outlined in figure 2.1

The number of radio cells is equal to the number of BTS. All the BTSs connected to a MSC have the same Location Area (LA). One or more MSCs form a unique LA, identified by a unique number called the Location Area Code (LAC). An example of a network topology with hexagonal radio cells can be found in figure 2.2.

The number of BSCs connected to a MSC is different for each operator. So the number of cell-ids that has the same LAC vary with each operator. The LAC topology of KPN and T-Mobile can be found in figure 2.3(a) and figure 2.3(b). The density of users determine the size of the LAC, for city areas the LAC will contain a few cell-ids whereas for the countryside a LAC can consist of many cell-ids, depending on the size of the cells. If a MSC has no capacity left a set of cells of that MSC is connected to a new MSC. A GSM cell is uniquely identified by a cell-id in combination with a LAC and a land-code or country-code. The term cell-id in this thesis refers to this unique combination.

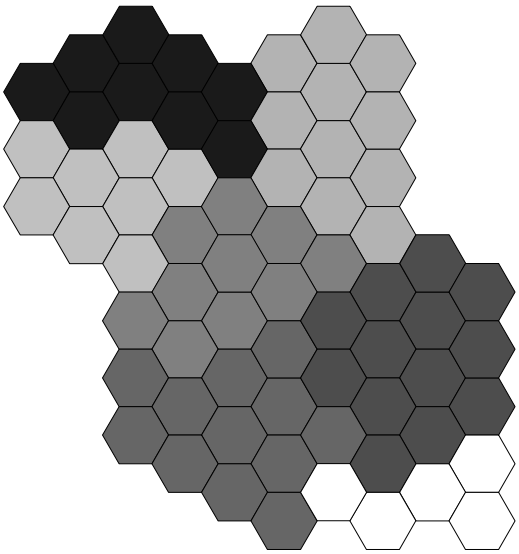
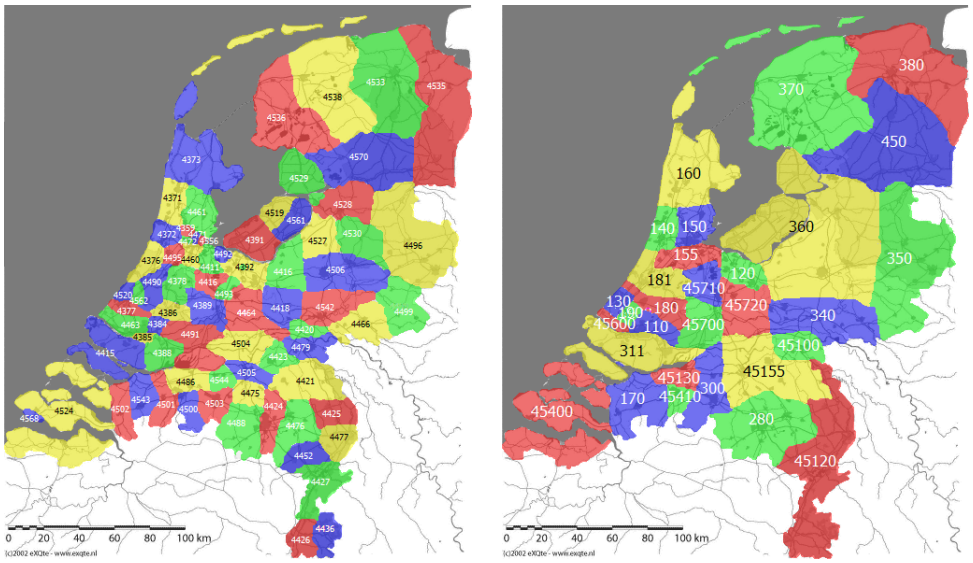


Fig. 2.2: Network topology: hexagonal view



(a) KPN (b) T-Mobile

Fig. 2.3: LAC topology

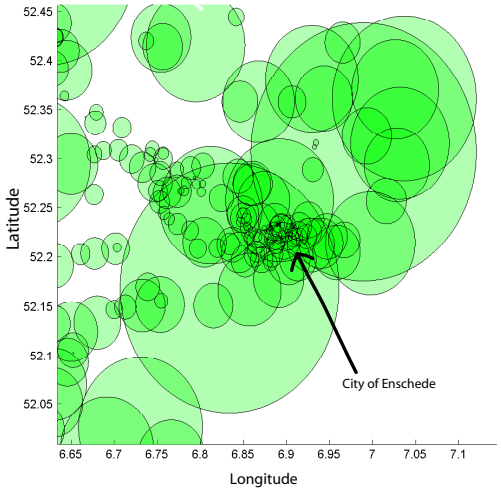


Fig. 2.4: GSM Cells: area of Enschede

An often used simplification of the network topology is represented as a concatenation of hexagonal radio cells in literature [NR04, HJ04a, LR05]. In practise this is not the case, in fact a lot of overlap is introduced to get a good coverage. The shape of a cell is not hexagonal but a polygon, shaped by reflections, distortion, influence of buildings, etc. Different people are creating databases with cell-id information gathered by mobile devices with GPS modules [KW]. From this data it can be concluded that a mobile phone can be in different cell-ids at the same place and overlap exists in the cell-id topology. A good example is the network topology for Enschede. Enschede is covered with three macro cells, i.e. cell size 1.5km till 15km, and a lot of normal cells, i.e. cell size 100m till 1500m, and pico cells, i.e. cell size 10m to 100m. The cell-id layer of Enschede can be found in figure 2.4 [KW].

The size of one cell is determined by the capacity needed in that area. For urban areas the size of a cell varies from 10 to 1000 meter. In country sides cells can be up to 15km. For example, in the Netherlands the maximum size of a cell that was found is 15km [KW]. For the area of Enschede (see also figure 2.4, Borne and Almelo the total number of different cell measurements is 863, see table 2.1. These measurements are from 1 Dutch operator, i.e. KPN. The database has not sufficient measurements to determine the diameters of every cell. The 184 cells with a diameter of 0 only have a few measurements and therefore have this 0 diameter. The other cells that have a diameter larger than zero have more than 10 measurements.

Diameter	Number	Diameter	Number	Diameter	Number
0	184	**4000**	48	**10000**	1
250	71	**5000**	7	**11000**	2
500	65	**6000**	18	**12000**	1
1000	158	**7000**	8	**13000**	1
2000	200	**8000**	2	**14000**	1
3000	93				

Tab. 2.1: Cell-sizes (diameter in meters) of area Enschede, Borne and Almelo (lat: 51.9-53, lon: 5.8-7.2)

2.1.2 GSM cell-id handovers

If a user has a conversation in a car on the highway with his mobile phone, he will drive through a lot of radio cells. The conversation must not be cut-off if the phone switches to a new cell. This switching betweens cells and channels when a phone call is in progress is called handover or handoff [Sch03].

There are four types of handovers:

- Switching channels in the same cell.
- Switching cells under control of the same BSC
- Switching cells under the control of different BSCs, but belonging to the same MSC
- Switching cells under control of different MSCs

The first two types of handover are called internal because they involve only BSC, and MSC is notified only on completion of the handover. The other two types of handover are called external because they involve MSC. For the proximity detection the last three handovers are of influence because they change the cell-id.

A handover may be initiated by the BSC or can by initiated by the MS. The MS scans the channels of up to 16 neighbouring cells, and forms a list of the six best candidates for possible handover. This list is updated to the BSC at least once per second. This information is used by the BSC and MSC for the so called handover algorithm. This handover algorithm takes the decision of switching to a new cell of let the MS connected to the current cell.

There are two basic algorithms for making handover decision for different criteria:

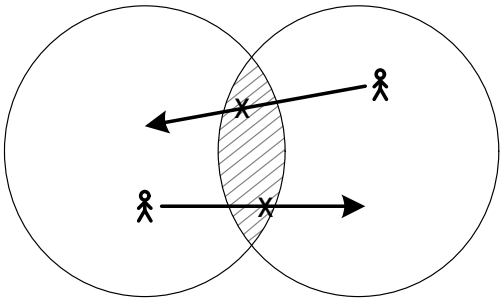


Fig. 2.5: Late cell-switches result in different cell sizes

- Minimum acceptable performance. If signal degrades beyond some point, transmission power is increased. If power increase does not lead to improve, handover is performed. Disadvantages: increasing transmission power may cause interference with neighbour cell.

- Power budget. Use handover to improve transmission quality in the same or lower power level. This method avoids neighbour cell interference, but is quite complicated.

A side effect of the handover algorithms is oscillation. If a mobile phone is in one place, it is still switched to the other cells. This happens because the signal quality changes. For example atmosphere changes, people interfere the signals; other mobile phones enter your cell or other input parameters of the handover algorithms change [osc05, KDK00].

This research is about proximity detection using the current GSM cell-id of a MS and it could be that proximity is detected between two users because of oscillation, because a user can oscillate to the cell of the buddy. If the oscillation is between two neighbouring cells, there is minor impact on the quality of proximity, in terms of proximity distance. The proximity distance will be less accurate because the size of cells can vary. In figure 2.5 is an illustration of two users moving to each others cell. Both users have a handover or cell switch on different places. Somewhere in the grey (the shaded part) area the users will switch from cell according to the handover algorithms. In figure 2.5 this happens at the x-marks in the grey area.

The second type of oscillation is when a user falls back into an overlapping macro cell. If this is the case and two buddies are in the same macro cell, the proximity distance between two users is large. This type of oscillation has a major impact on the detection of proximity and the accuracy of proximity distance.

	all CIDs	CID 46982	CID 46984	CID 47270	CID 13622
Average time in cell	0:27:58	0:34:55	0:25:58	0:07:36	0:08:38
Max time in cell	18:00:00	18:00:00	13:00:00	1:59:19	3:54:36
Min time in cell	0:00:04	0:00:13	0:00:04	0:00:17	0:00:18
Nr measurements	564	269	223	34	38

Tab. 2.2: Oscillation of a mobile device on one place

2.1.3 Oscillation experiment

An experiment was done on oscillation to illustrate the oscillation between neighbour cells, the results can be found in table 2.2. A GSM mobile device was placed on a spot where already different cell-ids were measured. The device was left for 7 days on the same spot and registered each cell-id change with time stamp. In these 7 days the device was connected with 4 different cell-ids. The longest time in cell-id 46982 was 3 days and 20 hours. This measurement is left out of table 2.2 to get a better overview. A total of 564 cell switches are detected in 7 days, with a minimum stay time of around 10 seconds and the longest time of 18 hours (without the 3 day measurement). The average time in a cell is around 30 minutes.

2.2 Proximity detection algorithms

This section first describes simple periodic algorithms to detect proximity. Then three relevant, more efficient, proximity detection algorithms found in literature are discussed, i.e. Strips algorithm by Arnon Amir et al. [AEM+04], Dynamic circles by Alex Kupper et al. [KT06] and Dead Reckoning by George Treu et al. [TWK06]. In the papers they do simulations to validate their algorithms. As location identifier of the mobile user GPS coordinates are used, the limits of GPS are not taken into account. For each algorithm the impact of using GSM cell-ids as location identifier is discussed.

2.2.1 Simple methods to detect proximity

Proximity detection is all about knowing the location of the users and calculate the distance between users to see if the users are in the vicinity. To keep track of the location of users the following update algorithms were found in literature:

- **Polling:** The location server requests the current position fix from the

mobile device. This may happen upon request from the application, on a periodic basis, or according to a caching strategy applied at the location server.

- **Periodic position update:** The device triggers a position update if a pre-defined time interval has elapsed since the last position update.
- **Distance-based position update:** The mobile device sends a position update if the line-of-sight distance between the last reported position and the current position exceeds a pre-defined threshold.

These update methods are periodic in time or in travelled distance. This will result in unnecessary location updates, e.g. if periodic position update is used the user will send updates regardless if he is moving, so the same location is updated several times. This is not efficient and these update methods are not considered for proximity detection using GSM cell-ids. The three more intelligent algorithms are discussed in the remainder of this section.

2.2.2 Strips-Algorithm

Arnon Amir et al. present an algorithm, called the Strips-algorithm to detect if a buddy is nearby [AEM+04]. To determine the location of a mobile user GPS coordinates are used. The evaluation of the algorithm is done using the IBM city simulator [Cit]. The strips-algorithm has the task to: 'Whenever a friend moves into the user's vicinity, both users are notified by a proximity alert message'. A friend is one that is predefined by enrolment or by matching the personal profiles of users.'

Let a, b be two users whose Euclidean distance denoted by $|a-b|$, is larger than R (R is the width of a strip). Let $l(a, b)$ denote the bisector of the line connecting a and b; i.e., the line consisting of all points of equal distance from a and b. $S(a, b)$ is the infinite strip of width R whose central axis is $l(a, b)$. See figure 2.6. This strip can be of different shapes, its desired properties are to have a *Hausdorff distance* $> R$. A Hausdorff distance is the maximum distance of a set to the nearest point in the other set. The idea behind this method is that as long as neither a or b_i enters $S(a, b_i)$, they do not need to exchange location update messages. If two users are notified with a message that they are within each others vicinity, a circle is computed around the two users. The radius of this circle is determined by $R/2 + 2 \times \epsilon$, centered at the midpoint of the line connecting the two friends. (ϵ is the desired tolerance for producing the proximity alert). If one of the users leaves the circle the distance between the users is calculated. When the distance is smaller than

the proximity distance, a new circle is computed, otherwise a new strip is calculated.

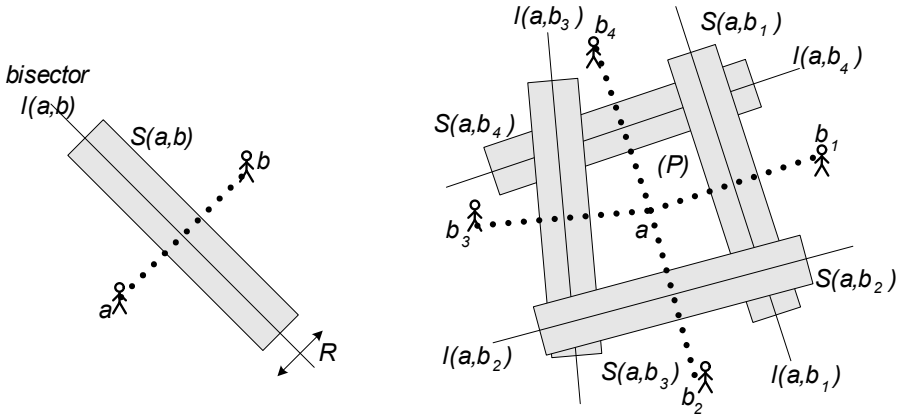


Fig. 2.6: Left: setting a new static strip of width R around the bisector between two mobile users. Right: user a does not need to update strips while moving inside the internal region (P).

The psuedo code of the Strips Algorithm can be found in Algorithm 1. $D(a)$ is the datastructure of a, it contains all strips $S(ab_i)$. e is the error of the location of the user.

```
SelfMotion()
while moving do
    a = ReadSelfLocation()
    Test(Distance(a))
    if a enters Strip(a,b_i) or MsgReceived(b_i) then
        StripUpdate(b_i)
    end
end
```

Algorithm 1: Strips: SelfMotion

To decrease the computational load of the mobile clients, they propose an approximated Strips algorithm. Instead of maintaining a convex region of possibly $\Theta(n)$ edges, for n friends, it maintains an approximated polygon of a fixed number of edges, at fixed, predetermined slopes. The region created by the polygon is called the internal region, denoted by (P). See right side of Figure 2.6.

`StripUpdate(`b_i`)`
send `a`'s location to b_i
receive b_i's location
if |`a`-b_i |< $R + \epsilon$ **then**
    `ProximityAlert(`*"b_i is nearby"*`)`
    `Delete(D(a), S(a,`b_i`))`
else
    `Compute S(a,`b_i`)`
    `Update D(a), D(`b_i`)` with `S(a,`b_i`)`
end

Algorithm 2: Strips: StripUpdate

2.2.3 Strips and cell-ids

If the strips strategy is used with GSM cell-ids it is hard to define a strip. This strip must be infinite long, so that users cannot enter the area of the other user without crossing the bisector. A simple formula can be calculated, representing the bisector, for each user when using coordinates, e.g. GPS latitude and longitude. With a cell-id topology it is harder to define a strip. First problem is the length of the strip. It could be shorter but then there is a possibility that the users cross the strip without knowing, i.e. moving around the strip. The second problem has to do with oscillation and macro cells. A user could switch to a macro cell that is not part of the strip and literally jump over the bisector. A solution could be to add all the nearby macro cells, and overlapping cells to the strip, but this will result in large sets of cell-ids for the strip. These sets have to be exchanged between the two users and make the strategy very inefficient.

Another issue with strips is the peer to peer part. For each buddy a user must keep a strip and calculates if he crosses the strip. With an increasing list of buddies the amount of resources used grows. The number of messages is equal to the number of users in your buddy group.

2.2.4 Dynamic Circles

Kupper and Treu present a method that looks like the strips-algorithm but they use circles instead of border regions [KT06]. They suggest some improvements to the algorithm and state that its outperforming the Strips-algorithm. In figure 2.7 each target is surrounded by a circular uncertainty area called A, which, from the location servers point of view, narrows down it's possible whereabouts. The circles are chosen in a way that, without reporting a position update, the mutual distance between two targets cannot

fall below the proximity distance d_p. The centre of the circle is the last known position of the user. The distance between the circles determines the size of the circle. This distance needs to be above the proximity detection distance.

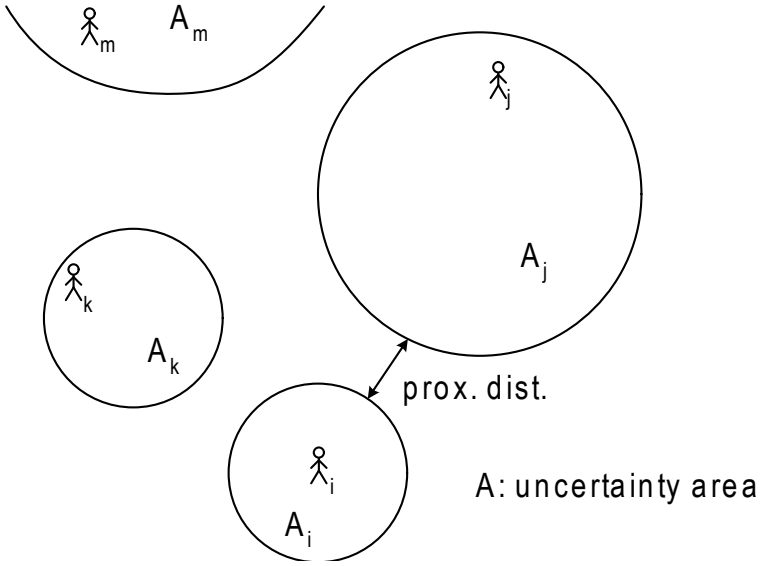


Fig. 2.7: Dynamic Centered Circles DCC strategy

2.2.5 Dynamic Circles and cell-ids

It is possible to define a set of cells to create a free region representing the circle in dynamic circles. But to find this set of cells, the knowledge of the topology is needed in the form of a graph. This graph must contain all the cells as vertexes and all the cell switches as edges.

2.2.6 Dead Reckoning

The newest proximity detection algorithm in literature is called dead reckoning. This algorithm uses the movement patterns of the observed targets to detect proximity [TWK06]. Based on a previously reported position that has been accompanied by additional information, such as speed and direction of the target, the mobile device and the location server simultaneously compute the current position of the device by a shared prediction function f. If the device detects that the so-derived position deviates from its actual position by more than a pre-defined threshold, it reports a position update. Several ways for computing f are known in the literature. True et. al. use the

direction and the speed of the mobile users to come to a shared prediction function.

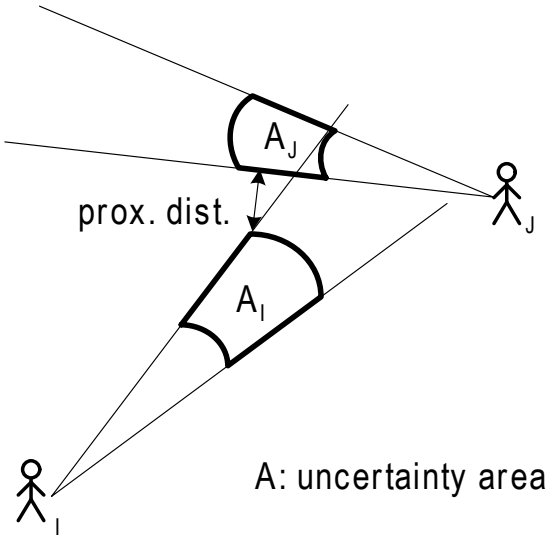


Fig. 2.8: Measuring the smallest possible distance between the uncertainty areas created by the proposed dead reckoning algorithm

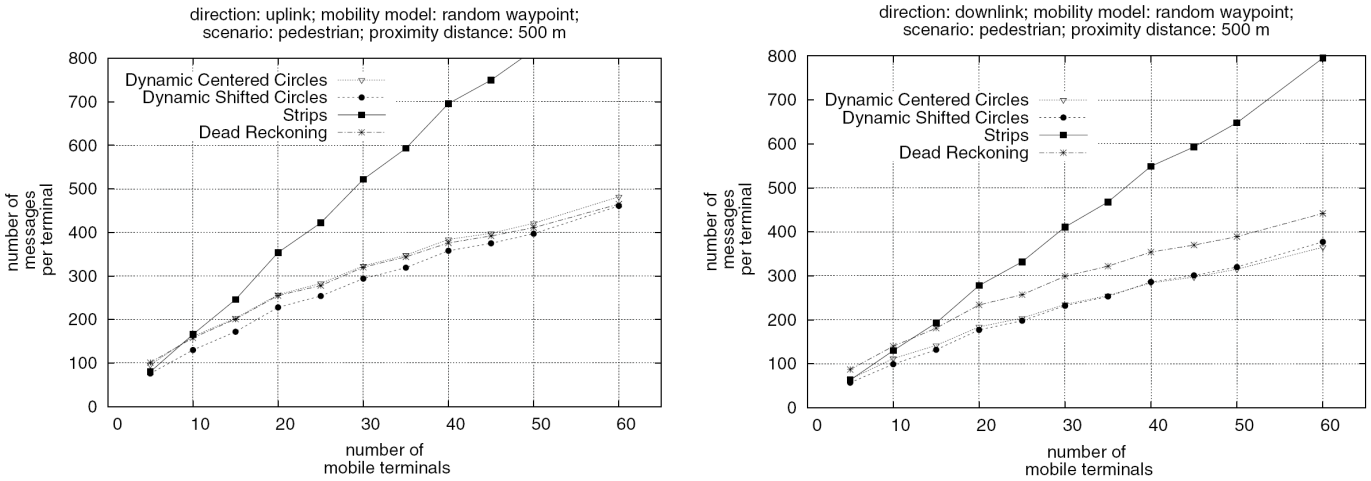
Figure 2.8 shows the computation of the smallest possible distance (spd) between i and j in a possible situation during execution of the strategy. The uncertainty area A_i of target i resembles a ring segment. The sector angle is determined by the direction of $\vec{v_i}$ and the angular threshold a, which is applied in both directions. The speed of the user determines the size of A.

2.2.7 Dead-Reckoning and cell-ids

Using dead-reckoning in combination with GSM cell-ids it is almost not possible to find a prediction function to predict the user's location. Without the knowledge of the network topology it is not possible to determine the future cells the user will be in. Meeuwissen et.al. [MRB07] have found, using the history of visited cells by a user, an algorithm that can approximate the Estimated Time of Arrival to predefined locations.

2.2.8 Comparison

The writers of the last algorithm, dead reckoning, have implemented the three algorithms in a simulator [TWK06]. All the algorithms use a latitude longitude as location identifier e.g., GPS, so the algorithms can use the same

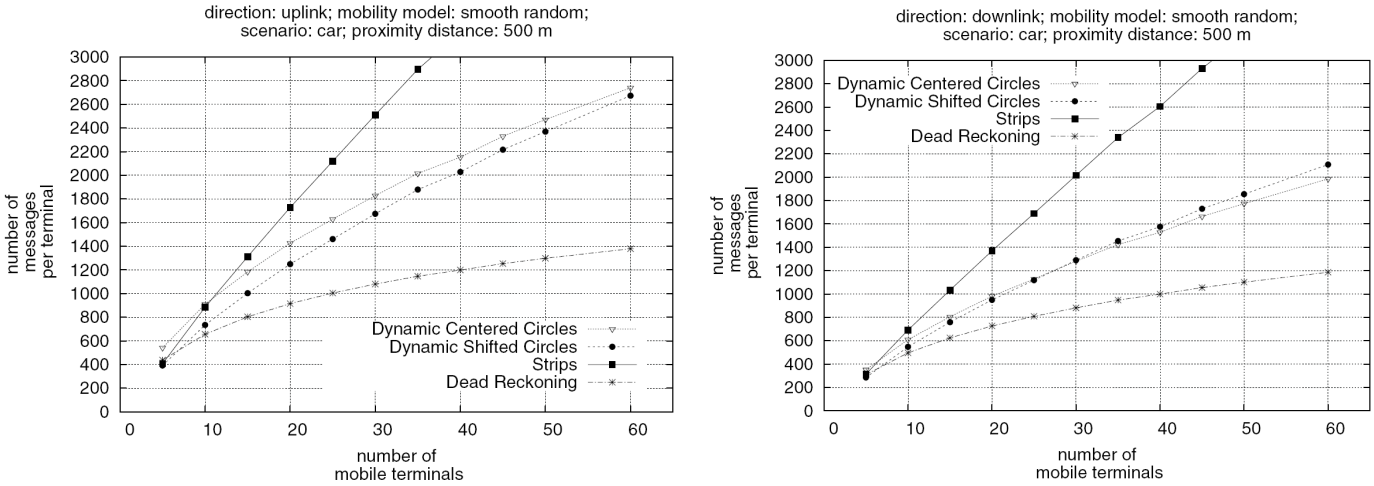


(a) Number of uplink messages

(b) Number of downlink messages

Fig. 2.9: Algorithms evaluated with pedestrian scenario [TWK06]

input data. The size of the simulation area is fixed and covers an area of $5km^2$. A proximity distance of 500m was used throughout the simulation. The number of users vary from 5 to 60 users and each user has the other users as buddies. The pedestrian was defined to have a maximum speed of 10km/h, which can be compared to a slow jog. However, his preferred velocities were set to 5km/h and 3km/h. The car scenario assumes a top speed of 55km/h and 8lkm/h to simulate slow rush-hour traffic. For different scenario's different algorithms are performing the best, though in general dead reckoning seems to perform best. In comparison to the strips algorithm the number of uplink messages are fewer but the number of downlink messages are increased (up and downlink with the server). Dead reckoning is outperforming the dynamic circles algorithm when increasing the average speed. The direction in which the users move determines also the quality of the algorithm. In Figures 2.2.8 and 2.2.8 pedestrian movement is used to evaluate the three different algorithms. For this type of movement the dynamic circle algorithms perform better than dead reckoning and strips. In figures 2.2.8 and 2.2.8 car movement is used to evaluate the algorithms. It is clear that dead reckoning is the algorithm that uses the lowest number of update messages per terminal when the average speed of the terminals increases [TWK06].



(a) Number of uplink messages

(b) Number of downlink messages

Fig. 2.10: Algorithms evaluated with car scenario [TWK06]

2.2.9 Algorithms and cell-ids

All these algorithms use a XY coordinate, e.g. GPS latitude and longitude, as location of the mobile user. It is hard to use these algorithms with GSM cell-ids as a location identifier. For strips it is almost impossible to define a strip using GSM cell-ids, it at least requires a lot of resources. Using the dead-reckoning strategy it is hard to define the user's direction and to estimate a travelling speed, because of the different sizes of cell-ids. The dynamic circles strategy is the most promising algorithm when using GSM cell-ids as location identifier. The idea of free moving regions can also be accomplished by using GSM cell-ids. The free region is then a set of GSM cell-ids. So the algorithms are hard to use one on one but are used as inspiration for the new algorithms using GSM cell-ids.

Conditions	Current	GPS	Backlight
program off; backlight off	26mA		
program off; backlight on	97mA		71mA
program on; GPS off; backlight off	28mA		
program on; GPS off; backlight on	98mA		70mA
program on; GPS on; backlight off	141mA	113mA	
program on; GPS on; backlight on	212mA	114mA	71mA
program on; GPS off; screen off	18mA		
program on; GPS on; screen off	130mA		

Tab. 2.3: GPS energy consumption experiments

2.3 GPS energy consumption

To illustrate the energy consumption of a GPS device we did an experiment on a high end mobile device, the HTC p3600. This device is a mobile phone with integrated GPS receiver running Microsoft Windows Mobile 5. The device supports several wireless connections, e.g. Quad band GSM, EDGE, Wi-Fi, Bluetooth 2.0 and is HSDPA ready [weba]. During the experiment only the GSM connection with General Packet Radio Service (GPRS) was on, all the other wireless connections were turned of. Several experiments were done with GPS on or off to get an illustration of the energy consumption of the GPS receiver. In table 2.3 it can be seen that if the GPS receiver is turned on the battery usage increases by a factor of 7. ACBPowerMeter [aw] was used to retrieve the Current in milli Ampère. Another tool was used to turn on the GPS and display the coordinates. If the tool is closed the GPS receiver is turned of. In the table this is denoted by "program off".

With a battery capacity of 1250mA the device will be empty in approximately 9 hours if GPS is turned on. With GPS off and no backlight the device has enough energy to run more than 70 hours. Both of the measurements are with the GSM phone capability of the device turned on. These measurements were done without making a call. According to the specifications the phone has a maximum call duration of 6 hours, this means that still if GPS is turned the battery will drain faster by, at least, a factor of 1.5.

Chapter 3

Algorithms

This chapter first explains three different deployments of algorithms, i.e. client to client, client to server and hybrid, using GSM cell-ids. Next, definitions are presented that are used in the explanation of the four proximity detection algorithms simulated during this research. These four proximity detection algorithms are client to server based deployed. The algorithms are divided into two groups, the Basic group and the graph Group. The Basic Group has two basic algorithms using only the available information from the GSM network. They are named, the Cell-ID algorithm and LAC algorithm. The Graph Group uses more information about the network topology in the form of a cell-id graph, representing each cell switch. This group contains two algorithms named the Graph1 and Graph2 Algorithm. At the end of this chapter additions are discussed and an alternative algorithm is given if a client to client deployment is used. The limitations and possibilities of this algorithm are discussed.

3.1 Different deployments

For proximity and separation detection between two or more clients, three deployments are possible:

- Client to client; or peer 2 peer, this can be compared with the serverless implementation of the Strips algorithm in section 2.2.2. Each client has the same responsibilities of the algorithm.
- Client to server; each client reports to an server and the server handles most of the responsibilities of the algorithms. Client only sends updates. This deployment can be compared to the dynamic circles algorithm in section 2.2.4.
- Hybrid method: The algorithm is divided over clients, servers and network.

In the related work of G. True and Kupper the client to server deployments are performing better than the client to client deployments of the algorithms under all conditions [AEM+04, TWK06, KT06]. This is due to the effect that client to client deployments generate more messages on an update. For example: if a client has 2 buddies, it has to send both buddies a location update, whereas with an client to server deployment the client only has to send 1 update message to the server. For this research also a client to server deployment is used. Simulations are done on four new proximity algorithms that use the GSM network to detect proximity and separation. Each algorithm uses different information of the GSM network to detect proximity and separation. We defined the following four new proximity detection algorithms:

1. **Cell-ID algorithm**; uses only the cell-ids available on the client to update to the server as location identifier. Location updates are done on each cell switch.

2. **LAC algorithm**; This algorithm uses the LAC and land-code also available on the client. Location updates are done on each cell, LAC or land-code switch.

3. **Graph1 algorithm**; The server has knowledge of the topology in the form of a cell-id graph. This is used to return a set of neighboring cell-ids to the client. Location updates are done if the client leaves the set of cell-ids returned from the server.

4. **Graph2 algorithm**; Comparable to the graph1 algorithm, but now the server returns larger sets of cell-ids instead of only the neighboring cell-ids of the client.

These algorithms send a proximity *notification* whenever users are in the same cell-id. As stated before this cell-id is unique combination of, cell-id, LAC and country-code. The algorithms do not send a 'goodbye' *notification* if users are no longer within each others vicinity.

3.2 Definitions used in strategies

The following definitions are used for all four strategies:

- **Proximity distance** is defined as the distance to detect proximity. In the related work with GPS solutions this distance is in meters, using GSM cells to detect proximity this distance is expressed in cells. For

the comparison of the four new introduced algorithms in this thesis, the proximity distance is one cell-id. This means that proximity is detected if two clients are in the same cell, with corresponding same cell-id.

- **Proximity detection** is defined as the detection that two mobile clients approach each other within a predefined proximity distance.
- **Proximity area** after proximity is detected two clients are put in a proximity area, to denote that those clients just had a proximity detection. For the four algorithms this is the cell-id were proximity is detected.
- **Separation detection** , whenever one of these clients leaves this proximity area, separation is detected. If this is the case both clients are informed that they are no longer connected.
- **Free-area** is the area in which the client can move without sending location updates to the server.
- **Location update** this term is used as an update of the location of a client in the algorithms. This is not the location update from the GSM network when switching to a new LA.

3.3 Cell-ID algorithm

The first algorithm to implement proximity detection uses only the cell-ids available on the mobile phone. As stated in section 2.1.2 the mobile phone receives a new cell-id upon a cell switch. On the mobile phone it is easy to detect this switching event. The client has the responsibility to update his new cell-id to the server upon each cell switch. See figure 3.1.

Upon a location update the server checks for proximity or separation. Four different scenarios can occur:

- There are no other clients in the new cell-id, no reply message is needed
- There is/are other client(s) in the new cell-id, send back a proximity notification
- The client is currently in proximity area and leaves the cell-id, clients are no longer connected and no action is needed
- The client is currently in a proximity area and leaves the cell-id and other clients are in the new cell-id, proximity messages are send and a new proximity area is formed.

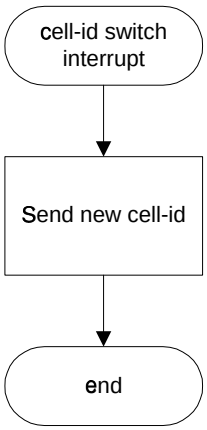


Fig. 3.1: client-side cell-id algorithm

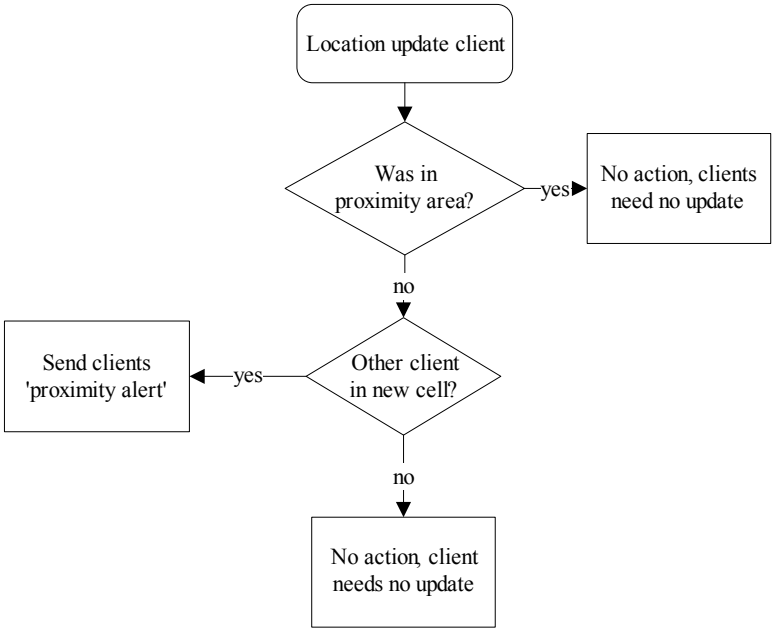


Fig. 3.2: server-side cell-id algorithm

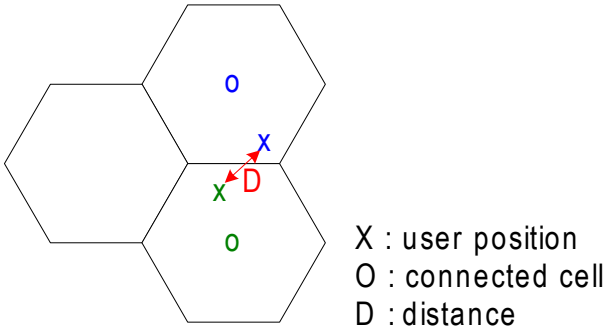


Fig. 3.3: Proximity missed

In figure 3.2 the working of the server-side of the algorithm can be found.

3.3.1 Limitations

The proximity distance is limited to the maximum size of a cell, because only the current cell-id is used in this algorithm. This means that proximity is only detected when clients are in the same cell-id. A consequence is that two clients could be close to each other but are in different cells. For an illustration of this problem see figure 3.3. The client positions are marked with an 'X', the distance between the two clients is D and the 'o' marks the cell-id the clients are in. The two clients are in a different cell but the distance 'D' between the clients is much smaller than the proximity distance, the size of a cell. So, although D is smaller than the cell radius, proximity is not detected, because the clients are not in the same cell. The flexibility of this algorithm is limited to a single cell proximity distance. It is not possible to have a proximity distance of 2 or more cell-ids.

3.4 LAC algorithm

The second algorithm uses the LAC and land-code of the GSM network, also available on the mobile phone. The algorithm is the same as the first algorithm but also checks if a client is alone in a LAC. If so the client only needs to update if he leaves the LAC and no longer on each cell-id switch. If other clients are in the same LAC the client must update on each cell-id switch. The same update technique is used when using land-codes. The proximity area can also be a land-code or LAC next to a cell-id. The proximity area default is a cell-id. The client has the responsibility to send only an update to the server respecting the current free-area of the user. This

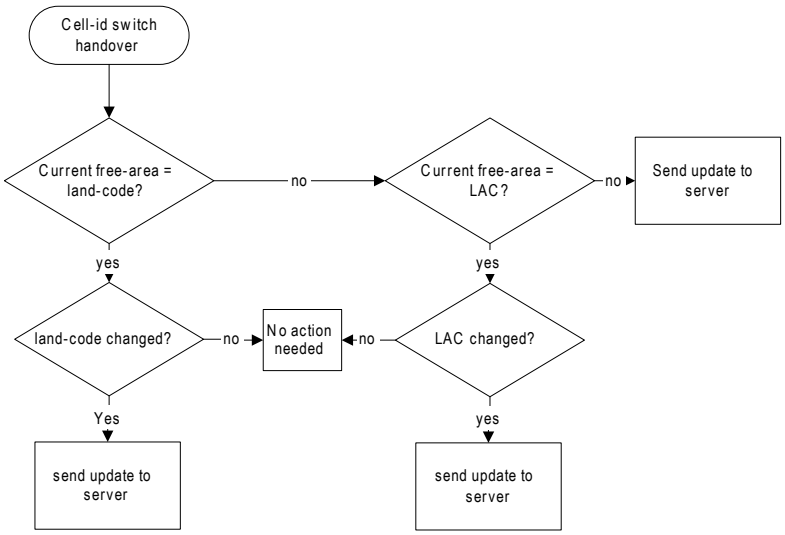


Fig. 3.4: client-side LAC algorithm

free-area can be a cell-id, LAC or land-code. The decision flow can be found in figure 3.4.

In figure 3.5 the flowchart of the server-side can be found. The client-side has the responsibility to update according to the free-area received of the server. If the free-area is a LAC, the client only needs to send an update to the server if the LAC he is in changes.

3.4.1 Limitations

The LAC algorithm uses more information about the cell-id and clustering of cell-ids than the first algorithm which only uses cell-ids. The advantage of the LAC algorithm is that if a client is alone in a LAC or land-code the client only needs to send updates upon a switch of a LAC or land-code, respectively. A disadvantage is that if another client enters your LAC the server has to update both clients, this result in more downlink messages in comparison to the cell-id algorithm. As can be seen in figures 3.5,3.4 the complexity of the algorithm increases on both the client and server side.

3.5 Graph1 algorithm

This algorithm uses the cell-id topology information as a graph. A neighbour of a cell is a cell in which you can come within one cell-id switch. This does not have to mean that cells are real neighbours, because macro cells create a

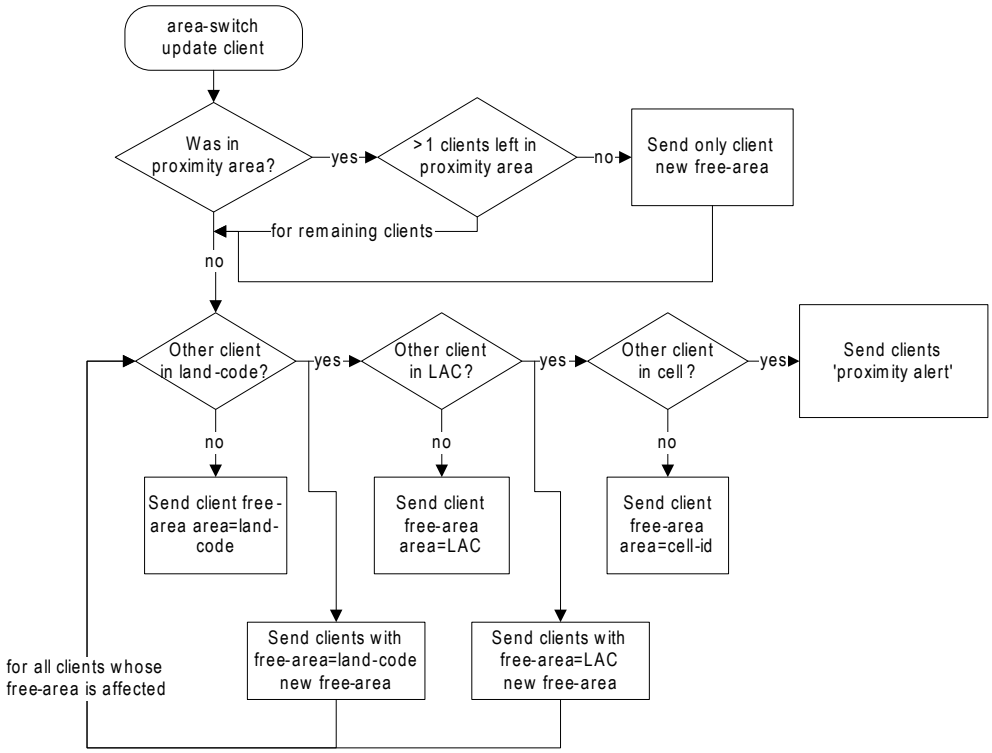


Fig. 3.5: server-side LAC algorithm

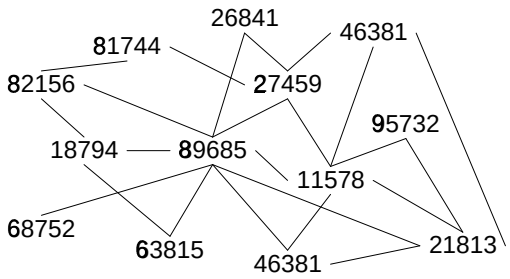


Fig. 3.6: Cell-id Graph, representing all cell-switches

lot of overlap it could be that there are multiple antennas between the two neighbouring cell antennas. In the cell-id graph each cell is a vertex and each neighbour of that cell is connected with an edge.

A graphical representation of this graph can be found in figure 3.6. Cell-id 89685 probably is a macro cell, because it has many neighbours.

Using this extra information of the network it is possible to create an algorithm with more possibilities. The proximity distance is no longer limited to one cell-id, as in the first two strategies, but could be multiple cells using the graph. The free area can be multiple cells. The server can detect if other clients are in the adjacent cells, of the current cell the client is in, if not these cells can be added to the free-area. It is also possible to state, with a certain probability, if a cell is a macro cell. This could give more information about the distance between two clients. If both clients are in the same cell that probably is a macro cell the distance between the two clients could be up to 15km (in the Netherlands).

The complexity of the algorithm on the client side is similar to the LAC algorithm. The client will receive a set of cell-ids from server in which he can move freely. The client has the responsibility to use this list on a cell-id switch to detect if an update to the server is needed. If the client is in a cell-id that is part of this free list it does not have to update otherwise the server is informed about the current cell-id. See figure 3.7 for the decision flow.

The free-area returned to the client consists of the new cell-id of the client and the neighbouring free cell-ids of this new cell-id. If a neighbour cell is already occupied this cell-id is excluded from the list.

The complexity of the algorithm on the server is much greater than with the previous two strategies. Upon an update the server has to lookup the cell-id in the cell-id graph and check for proximity, free areas and separation. In figure 3.8 the server-side of the cell-id graph algorithm is outlined.

If a client A enters the free area of client B the server will send only the

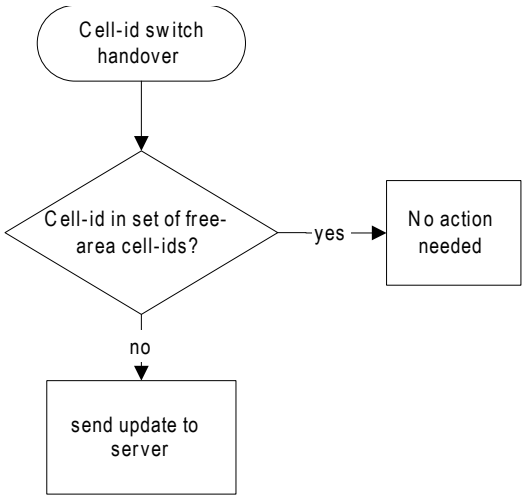


Fig. 3.7: Client-side: Graph1 algorithm

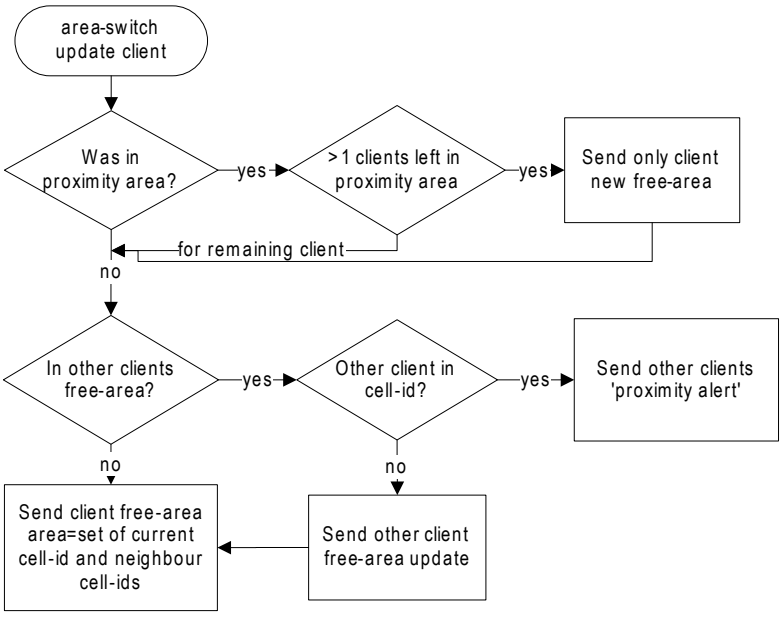


Fig. 3.8: Server-side: Graph1 algorithm

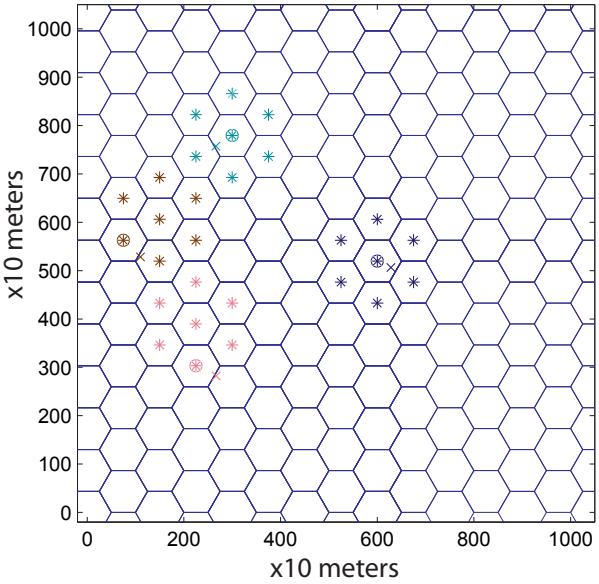


Fig. 3.9: Snapshot Graph1 algorithm

current cell-id back as free area for client A. The server also sends a message to client B to remove that cell-id from his free area.

For an example of a snapshot of the graph1 algorithm see figure 3.9. The '*' marks are the assigned cells to users, the 'x' marks.

3.5.1 Limitations

The algorithm is much more flexible, because it can support different proximity distances. The cell-id graph can be used to detect proximity if a client is multiple cells away. The cell-id graph however must be harvested from the network. This is a costly operation and changes if the operator decides to place a new antenna to get better coverage. It is also possible that an antenna is moved, the cell-id graph has to be updated.

If the density of clients is high, the neighbouring cells of a client are probably occupied. This result in a fallback to the first algorithm, the cell-id graph, on the client-side. The server still has to do the computations for the free-area per client.

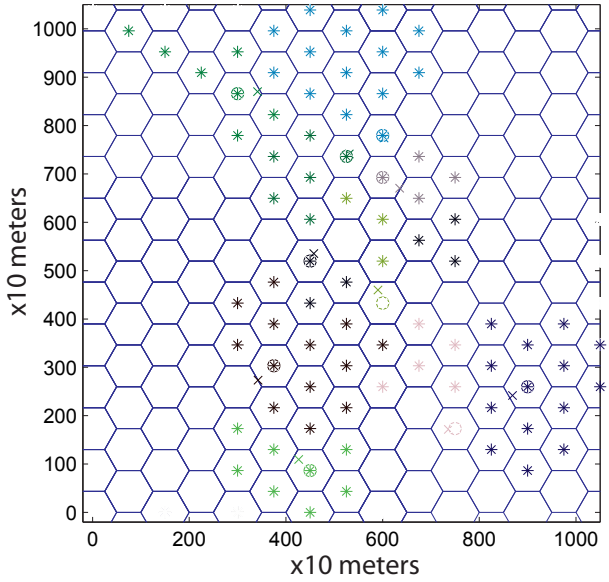


Fig. 3.10: Snapshot Graph2 algorithm

3.6 Graph2 algorithm

Using the idea of the graph1 algorithm, an extended version of the graph1 algorithm is found. The size of the free-area, so the set of free cell-ids, could be larger if the density, in terms of clients and number of total cells, is low. The client can move free in this set of cell-ids so a bigger set will result in fewer update messages to the server. The neighbour cells of the first ring of neighbour cells are also checked for free cells. For an example of a snapshot of the graph2 algorithm see figure 3.10. The '*' marks are the assigned cells to users, the 'x' marks.

3.6.1 limitations

A drawback of this algorithm is the number of server based updates to the user if a client enters another client's free-area. The free-area of that client has to be updated so the server needs to send a message to this client.

	Cell-ID	LAC	Graph1	Graph2
Uplink messages	−−	+−	+	++
Downlink message	++	+	+−	−−
Proximity distance	1	1	1+	1+
Client complexity	++	+	+	+
Server complexity	++	+	+−	+−
Resources	++	++	−−	−−

Tab. 3.1: Comparison algorithms: ++ is good, −− is bad

3.7 Summary algorithms

The cell-ID algorithm is very basic and the complexity is low. Disadvantage is that upon each cell switch an update must be send. Another disadvantage is that the proximity distance is limited to the same cell. The same holds for the LAC algorithm. The LAC algorithm will introduce less client-side updates but the server initiated messages increase. This is due to the fact that users will enter occupied LACs, the user owning the LAC has to be informed that he has to switch to a cell-id as free-area.

The graph group can have a proximity distance of multiple cells, using the cell-id graph. These algorithms have just as the LAC algorithm the disadvantage of generating more server initiated messages if a user enters another's free-area and no proximity is detected. This effect is even larger in the graph group. Each user has a free area, so the probability that a user enters another users free-area increases. See table 3.1 for a complete comparison of the four algorithms, in the table a ++ denotes an advantage of the algorithm and a −− a disadvantage.

3.8 Application improvement: Exclude clients

Using the cell-id graph it is possible, at the server side, to ignore buddies from a buddy list of a client, for an amount of time for example. This could be wishful if proximity is just detected between two clients and the client does not want to be notified for a day if that client is in the vicinity again. The server can than ignore this client after proximity detection, in the calculation of the new free area for the clients. This results in larger free areas, resulting in fewer location updates.

The server needs a mechanism to include the buddy again in the buddy group after a criterion e.g., a period of time or a distance based criteria. In

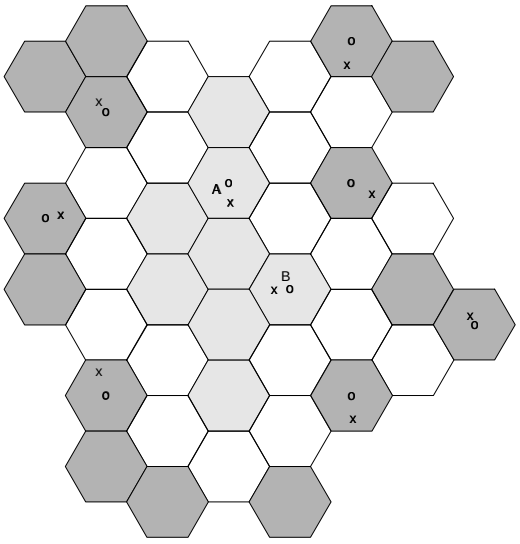


Fig. 3.11: Larger free areas possible because of exclusion of buddies

figure 3.11 an example is given after a proximity is detected between client A and B. The clients are separated and no longer in each other's buddy group, so the free-area for both clients covers also the cell-id of the other client. The proximity distance is a cell-id, the group of light grey filled cells is the free area of client A and client B. This improvement to the application is not simulated during this research, but will improve all four algorithms, because the buddy group is smaller.

3.9 Alternative algorithm: Peer to peer

The previous four algorithms, which were simulated for this research, are all client to server deployed. The client has only the responsibility to send location updates to the server, triggered by different parameters, e.g. a set of cell-ids. As mentioned before it is also possible to have another deployment, namely client to client or peer to peer. This section describes the advantages and disadvantages of using a client to client deployment to implement proximity detection using GSM networks. This client to client algorithm is not implemented and not simulated for this research.

With the cell-id graph it is possible to let the server find a set of cell-ids for a client in which this client can move freely, without sending location updates. With the previous strategies the server calculated this free-area using all the other free-areas of clients to create a non-overlapping free-area for a client. This results in a layer with clustered cells, representing the

free-areas. These clusters do not overlap for all clients.

Another solution would be to create free-areas for each pair of clients. This way it is possible to increase the size of the free-area for the clients, because with the calculation of the free-area only the free-area of the other client is taken into account. So for each buddy the client has to keep track of a free-area specific for that client. The client has now the responsibility of calculating the free-area for each buddy. So each client must have the cell-id graph. The maintenance and resources are difficult in this deployment. Updates of the cell-id graph must be sent to all clients instead of only to the server. The client must be able to store the cell-id graph.

The complexity of the proximity detection algorithm on the client side increases, because it now has to check on each cell switch if none of the free-areas are left. So if a client has a lot of buddies it has to check all those free-areas. If N is the number of clients, the client has to check N free-areas. To detect proximity the client must also request the cell-id of another client if the client enters the other client's free-area. So the responsibility of detecting proximity is shifted from the server to the clients. This result in more messages than with a client to server deployment, clients must request the cell-id of the other client to detect proximity.

So this algorithm can return much larger free-area than the previous strategies, this will result in less location updates of clients. The complexity is increased and the algorithm will be less scalable, on client, than the other deployments.

Chapter 4

Simulation Environment

For the simulation of the four algorithms from chapter 3 a simulator was developed in Mathworks Matlab [webb]. This simulator is developed to simulate the four algorithms with different mobility scenarios. This chapter describes the possibilities of the simulator.

First a diagram is given about the simulator that is used as guideline for this chapter. The diagram can be found in figure 4.1. The bold block is research done by other people. The inputs of the simulator are the mobility paths, generated by two java mobility pattern generators, i.e. CanuMobiSim and an extension on CanuMobiSim, VanetMobiSim [Steb, HF]. This is work done previously by people working at the university of Stuttgart and the Institut Eurecom and Politecnico di Torino and is explained in the following section. This generated data is parsed to a Matlab format and used as data to evaluate the algorithms. The additions to easily create input data or parameters for the simulator, e.g. parsers, batch files and tracing of real roadmaps, are not discussed in this thesis.

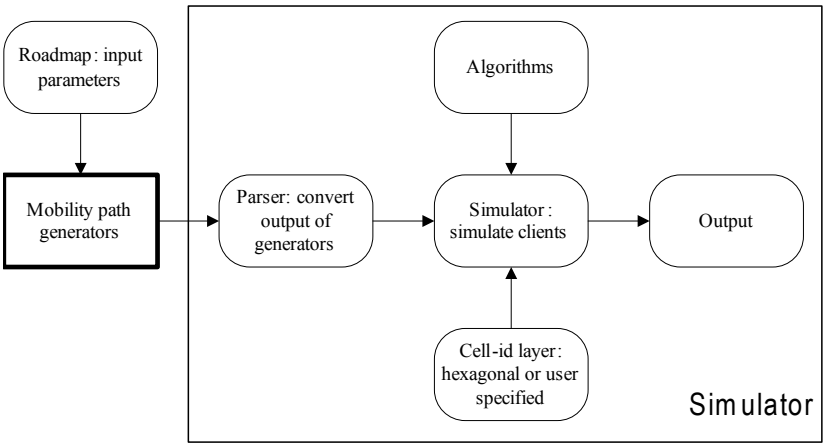


Fig. 4.1: Block diagram simulator

4.1 User mobility modelling

The first part of the simulator is the generation of the input data, the mobility patterns of users. To get realistic generated paths for the simulation a literature study was done on user mobility modelling. The best and free generator that had the necessary options, e.g. variable speed, road map, activity based patterns, was CanuMobiSim [Steb]. Jerome Harri et. al. have made an extension to CanuMobiSim that adds, among other things, vehicular models and the option to create own roadmaps, it is called VanetMobiSim [HF]. Among a lot of different mobility models, implemented in CanuMobisim and VanetMobiSim, the Intelligent Driver Motion and the Smooth Motion are supported. These two models are used in the literature often and are used for the simulation of the new four proximity detection algorithms. Both models generate car and pedestrian paths with velocities that have been adapted to closely fit with real deployment [HFB05].

It is also possible to log the movement of real users, using a GPS mobile device. This is realistic, but it is hard to harvest enough traces to get a complete dataset that covers most of the movement of users.

The input of the generators is in eXtensible Markup Language (XML). Simple tags are used to set variables, e.g. $< dimx >$ and $< dimy >$ for the size of the universe(generation area); and $< step >$ for the step size. This step size determines the amount of maximum milliseconds between each step of a user.

VanetMobiSim adds the Spatial Model to CanuMobiSim and has the option to generate or create your own roadmaps. A graph in the generators represents this roadmap. This random graph is built by applying a Voronoi tessellation(this is a special kind of decomposition of a metric space determined by distances to the specified discrete set of nodes in the universe) to a set of randomly distributed points. It is possible to define areas (clusters) with different node densities, creating a non-homogeneous graph. For this research a tool is also implemented to trace a roadmap and generate the XML data for a user defined graph. The mobility generators use this graph(roadmap) to generate the paths.

The output of these generators is a text file. This text file contains for each user a time and location coordinate. The time is increasing with a predefined step and indicates the user's position at that time. This output data is used as input data for the simulator, written in Mathworks Matlab. For an example of the output data see table 4.1

Although the generators produce realistic movement for users, there is no model implemented for day-to-day trips of humans. So it is not possible

```
$node_(19) set X_ 746.000001
$node_(19) set Y_ 954.000001


$ns_ at 0.0 "$node_(10) setdest 652.8500009928754 371.000001 0.3"
$ns_ at 0.0 "$node_(11) setdest 285.1460135821853 889.0343569016907 0.3"
```

Tab. 4.1: output example mobility path generator

to generate the movement of a worker for example or of a housewife each with their typical activity patterns. It is possible to produce trips according to automaton of activity sequences in specific mobility models. The trips are then produced between points of movement area belonging to spatial environment. For path searching, Dijkstra shortest-path algorithm [Dij59] or Dials STOCH algorithm [Dia71] can be used. So it is possible to implement a model that generates day-to-day real life paths but it is not included in the generators. We did not implement this model and use only the available models in the movement generators during this research.

Another limitation of the tool is the excessive use of resources. The generators are implemented in Java, and use a lot of memory and CPU if larger universes are defined. If the number of users in the universe is increased the amount of resources used also increase. This is a problem in the simulation of the use cases with a large area, it is not possible to generate mobility patterns for 100 users in a $10km^2$ universe. A solution to this problem is to set the universe to $1km^2$ and decrease the speed of the users in proportion to the virtual universe of $10km^2$. The last step to create this virtual universe is to set the amount of cells on the area as if it where a cell topology for $10km^2$. The step size of the users is far less than the size of cells, so no inaccuracy is introduced using this workaround. Increasing the step size could be an alternative solution but the size of the simulation area determines the memory usage. With the first solution the memory used by the generator stays the same.

4.2 Simulator

For the comparison and evaluation of the algorithms a simulator has been developed using Mathworks Matlab. The simulator consist out of five main

parts: (see the simulator part in figure 4.1)

1. Parsers; to parse the output files of the generators and the logged data
2. Simulator; the heart of the simulator, reads the paths and simulates the users in time
3. Layers; cell-id layer, adds a generated or gathered cell-id topology
4. Viewer; the GUI of the simulator, displays the users movement and cell-id topology
5. Algorithms; the algorithms for proximity detection

The parsers can parse different formats of raw user movement data into one single Matlab ready format. The cell-id layer can be a generated topology consisting of hexagonal cells (size is variable) or be a close to real cell-id topology generated with information of the database of Telematics Institute [KW]. To follow the working of the algorithms a viewer is designed that shows the users movement in time. The speed is adjustable and the tracking of users is optional. The algorithms can use the data available in the simulator, e.g. the user's location and the cell-id layer.

The real location of the users in terms of x,y coordinates is available in the simulator. When proximity is detected using GSM cell-ids it is possible to compare the real proximity distance with the distance measured in cell-ids. The accuracy of the proximity distance measured using GSM cell-id topology can be calculated. The real location of the users can also be used to detect false positives and false negatives. A false positive is that there is no proximity but a proximity notification is sent to the users and a false negative is a missed proximity notification.
To sum up the following variables can be gathered during the simulation.

- Number of messages sent from the client to the server
- Number of messages sent back to the clients from the server divided in:
  - Location update messages, server assigns new free area for a users
  - Proximity area messages, server assigns two users with a separation area
  - Proximity detection messages, notify users in each others vicinity
- The total number of proximity detections (For realistic topology):

– The difference in proximity distance of the real distance and the maximum distance that can be calculated using the diameters of the cells.

– The missed proximities or false negatives, for a predefined proximity distance in meters.

4.3 Handovers and Oscillation

The simulator can also simulate the handover or cell-switch between two adjacent cells. A grey area is defined by creating overlap between two adjacent cells. In this grey area the user can switch to the other cell. The probability of switching is equally distributed in this grey area. It is hard to model the handovers/cell-switches, because a lot of variables are used in the algorithms that decide the handovers, e.g. atmosphere, power and if you are in a call. In figure 4.2 the points where users switch from cells are marked with a $*$.

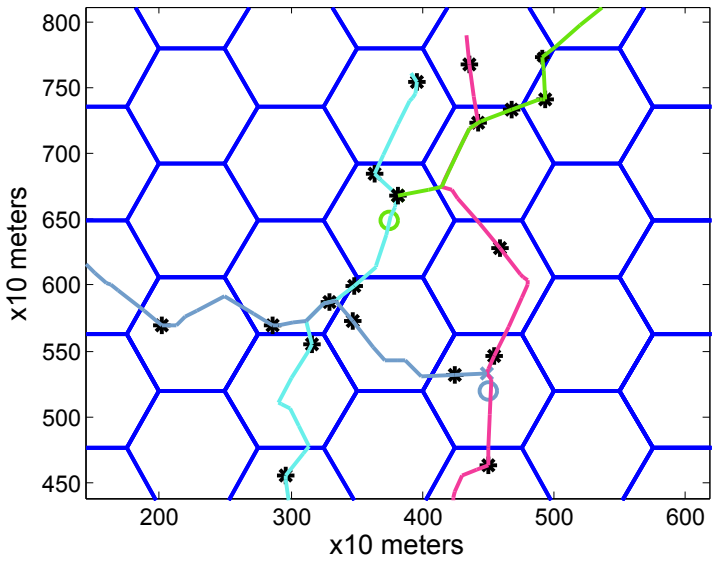


Fig. 4.2: Cell-switch position of users marked with $*$

Oscillation is also implemented using a timer with a random variable. If a user is in a place where two cells overlap a cell-switch can occur to the other cell and back. The timer is set for each user.

4.4 Real user paths

Next to the generated movement of users, the algorithms are validated with real user logged paths. Two devices are used to gather the data, the HTC P3600 with onboard GPS receiver and the Qtek S200 with Bluetooth GPS receiver. A small application gathers the cell-id switches and the GPS coordinates (interval of 3 seconds) and stores the data in a file. This file can be used as input for the simulator using the parsers. To get proximity detection at least two different persons have to log their paths. For this research the group of people logging their paths are all people having a 'nine to five job' and are colleague that don't live in each others neighbourhood. Therefore the data gathered is mostly limited to proximities at work or at meetings in other cities. Some paths of people are time shifted to create proximity detections, if this is the case it will be mentioned in the evaluation section and explained what kind of time-shift is done. An example of a path of two users can be found in figure 4.3. The x-marks are the logged path of one user and the 0-marks of the other. The different colours are used to indicate the cell switches.

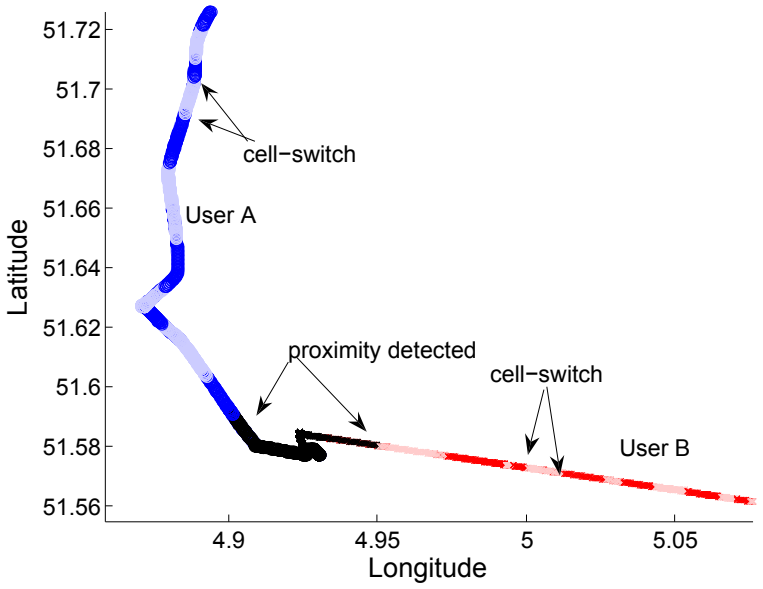


Fig. 4.3: Two logged user paths approaching eachother

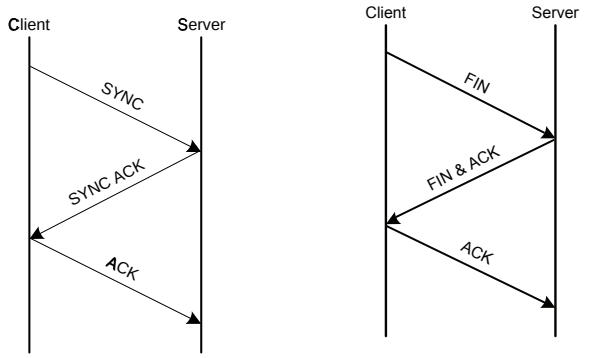
CS	1 slot	2 slots	3 slots	4 slots	5 slots	6 slots	7 slots	8 slots
CS-1	9.05	18.2	27.15	36.2	45.25	54.3	63.35	72.4
CS-2	13.4	26.8	40.2	53.6	67	80.4	93.8	107.2
CS-3	15.6	31.2	46.8	62.4	78	93.6	109.2	124.8
CS-4	21.4	42.8	64.2	85.6	107	128.4	149.8	171.2

Tab. 4.2: GPRS data rates in kbit/s per Coding Scheme [Sch03]

4.5 GPRS traffic

The simulator has an implemented Transmission Control Protocol / Internet Protocol (TCP/IP) traffic model to calculate the data traffic over GPRS per user. First some background is given about GPRS and than is explained how data traffic is calculated using TCP/IP. The messages between the client and the server will be sent via TCP/IP over GPRS. GPRS can offer efficient wireless access to external IP-based networks, e.g. the internet. The main benefit for users of GPRS is the 'always on' characteristics, no connection has to be set up prior to data transfer. The transfer rates are specified by the available time slots in TDMA and the Coding Scheme. The coding scheme is chosen by the system depending on the current error rate. Today, a typical MS is a class 10 device using CS-2, which results in a receiving rate of 53.6 kbit/s and a sending rate of 26.8 kbit/s. See table 4.2 for a complete overview of the data rates.

For the simulations devices of class 10 are used, resulting in 4 receiving slots and 2 sending slots with a maximum of 5 slots. [Sch03]. TCP/IP will be used as transmission protocol. The messages are divided in uplink and downlink messages. The TCP and IP header are both 20 bytes and are the overhead of each packet with a maximum data size of 1500 bytes. To set up a TCP/IP connection a synchronisation must be done. First a SYN(synchronisation) message is send to the server from the client, the server will respond with an SYN acknowledge message and the client will return an ACK (acknowledge) message. It is possible to send the data with the last ACK. See figure 4.4(a) for the handshake. With this background information the data traffic can be monitored per user. If a client wants to send a location update it will cost $3 \times 40 = 120$ bytes for the synchronisation and 40 + data-bytes for sending the packet with the data. If the server sends a reply message it will sent acknowledge or a packet with data. To respect the TCP/IP protocol the connection has to be closed. This can be done by sending a packet with FIN flag set on. This results in an acknowledge message from the other side, so 80 bytes of traffic are used to close a TCP/IP connection.



(a) TCP Synchronisation (b) TCP Termination

Fig. 4.4: TCP/IP

The server side has to close the connection with the same handshake so another 80 bytes are used. See also figure 4.4(b). If the connection is made the information/data needed for the algorithms can be sent. As stated before a GSM cell is uniquely identified by a cell-id, LAC, and land-code. Each of these variables are 2 bytes in size in the simulation. In the third and fourth strategy this will result in bigger packets, because multiple cells are send back to the client from the user representing a free-area.

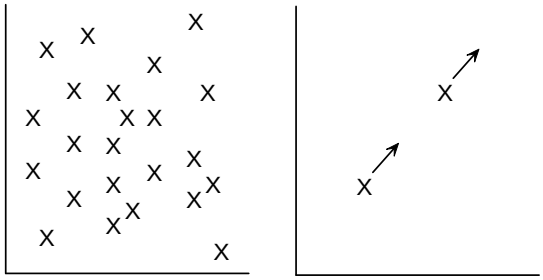
4.6 Simulation scenarios

The simulation input set is defined according to scenarios, different movement of users and special scenarios. The scenarios are covering the major part of user movement and are used in the latter phases of the research to evaluate the algorithms on different areas.

Using the generators it is possible to assign minimum, maximum speeds and acceleration to groups of users. For this research three categories of users are defined(all ground-travel), the speed as well as the acceleration within each category is variable:

- Pedestrians (3-5km/h)
- Bikers (15-20km/h)
- Car/train(fast travel) (60-120km/h)

When simulating all three categories of users is mixed equal, 1/3rd of pedestrians, 1/3rd of bikers and 1/3rd fast moving traffic. A buddy group



(a) Distributed (b) Chasing

Fig. 4.5: The main scenario and a special scenario

then will contain pedestrians, bikers and cars. If the buddy group is not modulo three the rest of the buddies are simulated as cars.

The main scenario is the distributed scenario. In literature this scenario is used most of the time [TWK06, KT06, AEM+04]. This scenario calculates a random initialisation position (see figure 4.5(a))and let the users walk through the whole simulation area. For this research the Intelligent Driver Motion is used as mobility model for the distributed scenario. This model is comparable to the Smooth motion with an extension to adjust speed of cars to avoid collisions. For a realistic simulation 1/3rd of the users are simulated as pedestrians, 1/3rd are bikers and the rest is simulated as if the users were in a car. This scenario embeds different movement of users:

- Movement within a group of other users
- Movement out of a group of users and vice versa
- Travel between a group of users
- Travel in a group

For this research we expect strange behaviour in special scenarios, because the cell-id layer is used to detect proximity. One special case is simulated for this research to see more details of the differences between the four algorithms:

- Two users chasing/following each other. See figure 4.5(b).

Chapter 5

Simulation

This Chapter describes the simulations done on the four new introduced proximity detection algorithms using the GSM network. The simulator of the previous Chapter 4 is used. First an explanation and example is given about the confidence level and interval. The confidence level and interval are used to state that the simulations are reliable. The accuracy of GPS versus GSM cell-ids is simulated using a realistic topology extracted from the telematica database [KW]. A distributed scenario, often used in literature, is used to scale the number of users and get the differences in performance, in terms of number of messages and data traffic, of the four algorithms. A typical scenario is used to show the effects of chasing clients on the algorithms. An illustration is given about the amount of data used, using the distributed scenario. At the end of the chapter a simulation run is illustrated using real logged data.

5.1 Confidence level and interval

Because randomly generated paths, according to mobility models, are used for the simulations a confidence level is chosen and confidence interval is calculated. This is done to get a reliable simulation. In literature a confidence level of 95% is often used [HJ04b]. With 95% probability the real mean is within the found interval, which is typically the mean of different runs $\pm$ a relative precision ϵ. Usually the relative precision of the estimators is chosen to be 0.05 or 5% of the estimated values.

To come to a confidence level of 95% a large number of runs ($n > 30$) must be done, using the same parameters except the random placement and movement of users. If the number of runs is less than 30 it is possible that the result is biased. For each simulation run in this chapter the confidence level and interval is calculated, to get a reliable run.

A $100(1 - \alpha/2)$ confidence interval is calculated as follows [HJ04b]:

$$\bar{X} \pm z_{(1-\alpha/2)} \frac{\sigma}{\sqrt{n}} \tag{5.1}$$

Where $\bar{X}$ is the mean of the samples in set X. σ is the standard deviation of the set of samples. The z-factor or the $(1-\alpha/2)$-quantile of a unit normal variate is selected from table A3.5.1 in [S.M87]. For a confidence level of 95% ($\alpha = 0.05$) the factor $z_{0.975} = 1.96$.

To illustrate the calculation of the confidence interval an example is given that calculates the confidence interval over a set of 50 simulation runs. One simulation run has a length of 200 hours and simulates the user movement of 5 users. Each user has 4 buddies. The area of the simulation is a middle sized area of 15km by 15km with a suburban roadmap. The average number of proximity detections per user per run of 200 hours can be found in figure 5.1.

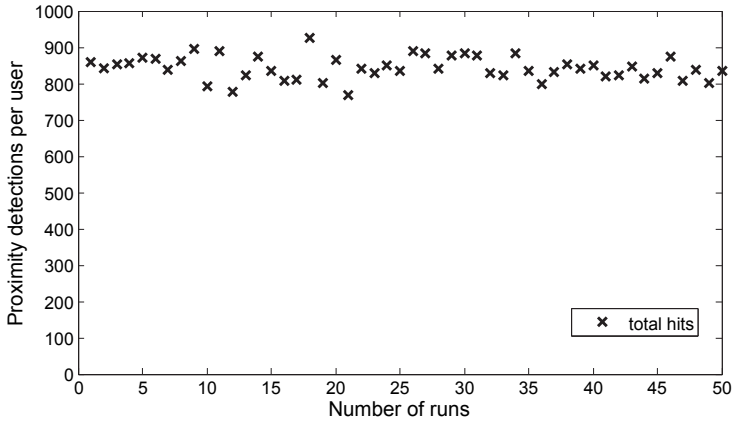


Fig. 5.1: 50 simulation runs of 200 hours on a city area of 15x15km

The mean $\bar{X} = 920$ of $n = 50$ runs. $\sigma = 40.5514$ Filling in the equation 5.1 results in the following confidence interval in which the real mean is found with 95% probability:

$$920 - 1.96 \times \frac{40.5514}{\sqrt{50}} \approx 909 \tag{5.2}$$

$$920 + 1.96 \times \frac{40.5514}{\sqrt{50}} \approx 931 \tag{5.3}$$

To see if the set of samples follows a normal distribution the normal probability plot can be found in figure 5.2. The points in figure 5.2 form a

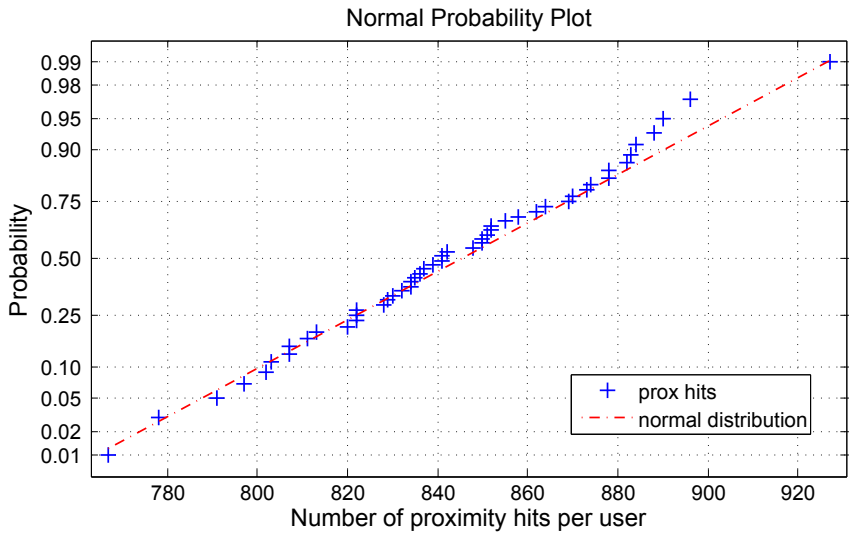


Fig. 5.2: Normal probabiltiy plot of samples

nearly linear pattern, which indicates that the normal distribution is a good model for the simulation data set.

In the simulator the calculation of the confidence interval is done 'on the fly'. This means that the simulation is terminated if the desired precision is reached, with respect to the minimum of 30 runs. See figure 5.3 for the process of simulation. Because the part that generates the mobility part is independent of the simulator one replication is one simulation run of multiple hours of user movement data, in the example above it was 200 hours per run. This results in differences in the calculation of the confidence interval per simulation, due to the granularity. In the simulation Graphs the confidence interval is displayed in the title of the figures. For example if the Conf. Int. is $< 2\%$, this means that for all the simulation points in the graph the confidence interval is lower than $\pm$ 2%.

5.2 GPS versus GSM cell-id accuracy

Using the cell-ids from 'Telematica Instituut' it is possible to create a realistic cell-id layer. The data from the database of the area of Enschede is not complete, but it could be used to create a layer via extrapolation. To create this layer, first the dubious measurements are filtered out, i.e. cell-ids with a zero LAC and the cells with 10 or less measurements. After this filtering only 338 of the 863 cells are left over. This layer has a lot of white spots where no coverage is, this is not realistic, because in the Netherlands almost

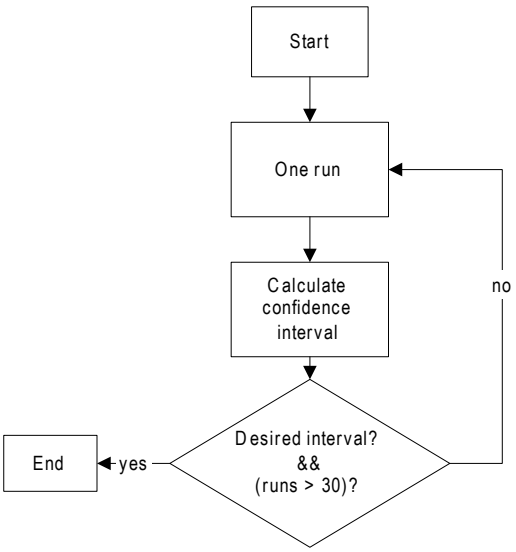


Fig. 5.3: Simulation process to confidence interval

everywhere there is coverage. To get a realistic layer that has coverage, a part of the cell layer (with coverage) is taken out of the filtered data and copied several times to create a larger layer. This new layer can be found in figure 5.4. We are focusing on the simulation scenario of an urban environment.

The simulation area is 1000x1000 units (1000x1000 for faster simulations) in figure 5.4, so the cell layer is larger than the simulation area. This is done to get a realistic situation, i.e. to avoid strange behaviour at the borders of the simulation area. This area can be compared to 25x25km in real life. If a user is in a white spot of the layer, where there is no coverage of cells, the user is not connected to a cell. This is done to keep realistic diameter size of cells. Therefore no proximity will be detected in any of these white spots. After a run of 100 users of 10 hours using the 'Intelligent Driver Motion' model of VanetMobiSim [HF] 23780 proximity detections are simulated. For each of these proximity detections the diameter of the cell (the cell where proximity is detected) and the real distance between users is stored. Since the cell-layer is used to detect proximity users can only be informed with an accuracy corresponding to the diameter of the current cell where proximity is detected. This means that if proximity is detected between two users the maximum distance between those users is the diameter of the cell, in which the proximity is detected. Proximity must then be detected if two users are in the same cell.

In figure 5.5(a) a histogram can be found of the differences between the real distance between two users and the distance that is found using the

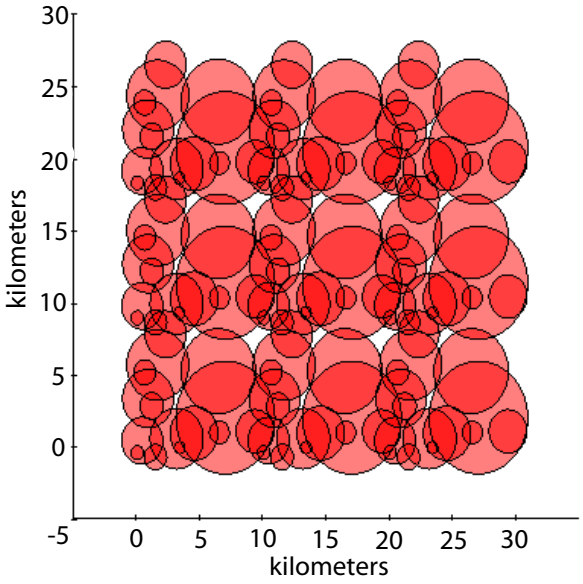
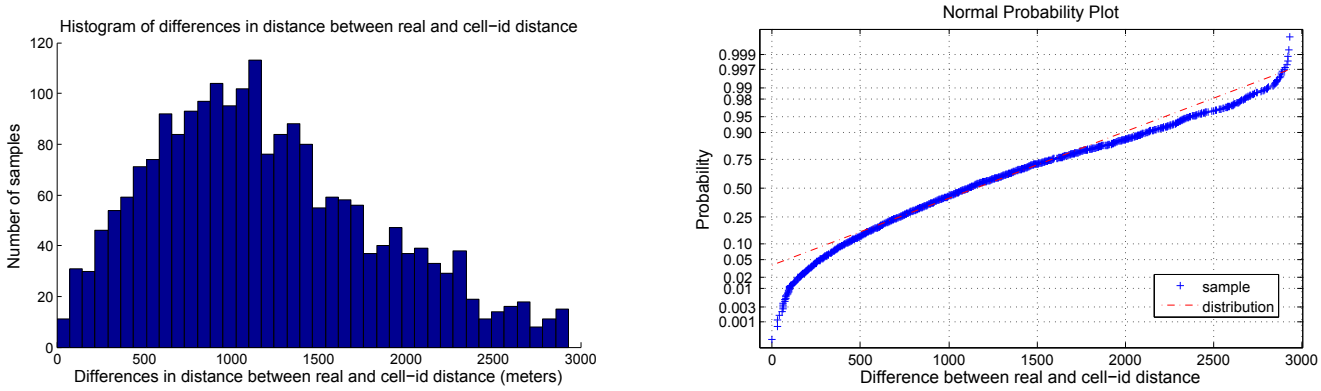


Fig. 5.4: Extrapolated realistic cell-id layer (25x25km)

diameter of cells. These distances are calculated upon proximity detection between those users. In figure 5.5(b) the normal probability distribution can be found of these differences. It can be seen that the maximum difference between the real distance and the cell distance is around 2900m and the smallest difference 0. The difference can be zero if both users are at the edge of a cell. The average difference is approximately 1100m.

The accuracy of proximity distance using GSM cell-ids is limited to the diameters of the different cell-sizes as mentioned before. It is possible that two users are within a certain range of ,for example, 500 meters and proximity is not detected using the GSM cell layer. This is called a missed proximity, also often called a false negative. The realistic cell-id layer is used to illustrate the number of missed proximities, see table 5.1. This table displays the total missed proximity detections. The total number of proximity detections during this run was 11800 and the total number of users 50. The simulation area is an area of 25x25km. The average size of the cells is approximately 1100m. Around this proximity distance the number of missed proximities is 41 or 3.5% of the average total number of proximity detections.



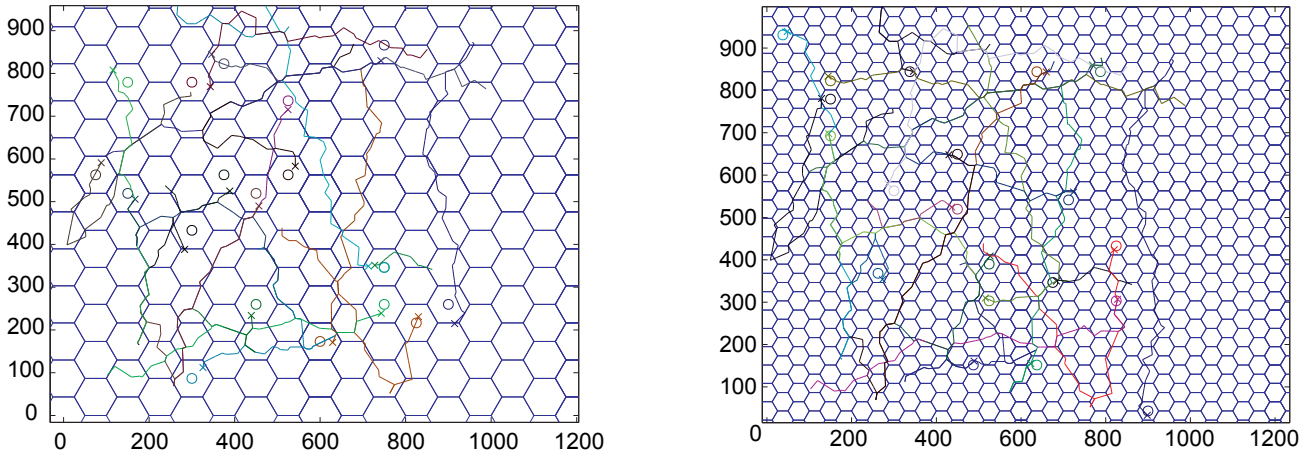
(a) Histogram differences

(b) Normal probability distribution

Fig. 5.5: GSM proximity distance differences

Distance	Missed proximity	Percentage(1180)
900	258	21.8%
1000	92	7.8%
1100	41	3.5%
1200	28	2.4%
1300	26	2.2%

Tab. 5.1: Missed proximities per user per 10 hours



(a) medium: 140 cells (10x10km)

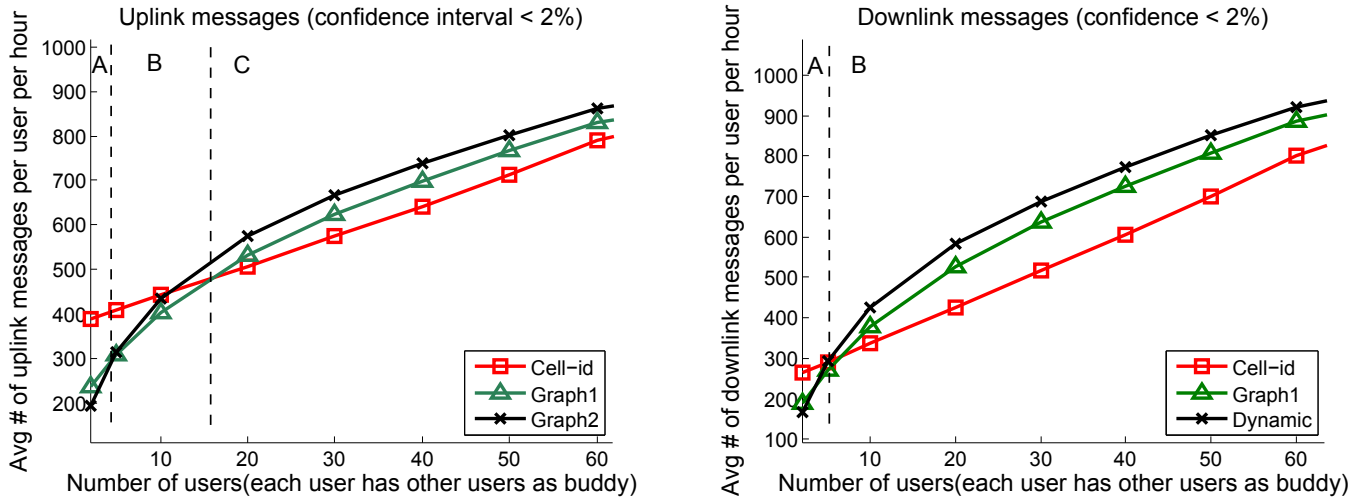
(b) large: 680 cells (25x25km)

Fig. 5.6: Distributed scenario large and small simulation

5.3 Distributed scenario

To compare the different algorithms, a run is done with distributed users. The total number of users is scaled in the same simulation area. This scenario is mostly used in literature to compare proximity detection algorithms [AEM+04]. A voronoi graph is used as a roadmap. The clusters in the voronoi graph are representative for an suburban area [Cho97]. Two simulations are executed with different simulation areas. A medium area of approximately 140 cells (representing 10x10km) and a large area of 680 cells (representing 26x26km). The two simulation areas can be found in figures 5.6(a),5.6(b), these are snapshots during a simulation run. The users are the 'x' marks and the lines are the corresponding paths of the users. The hexagonal grid represents the cell-id layer. The 'o' marks in the centre of the hexagons denote the current cell the user is connected to.

The four algorithms, Cell-id, LAC, Graph1 and Graph2, are simulated with the distributed scenario input data. First the medium size simulation is evaluated (see also figure 5.6(a). The LAC algorithm is left out of the figures of the medium size simulation, because it performs the same as the Cell-id algorithm. In the medium area the number of LACs is 1 or 2, this is too low for the LAC algorithm to have an advantage. As seen before in section 2.1.1 the number of cells that form a LAC are more than 40 cells, only the area of Amsterdam for the KPN topology has fewer cells in a LAC.



(a) Uplink messages

(b) Downlink messages

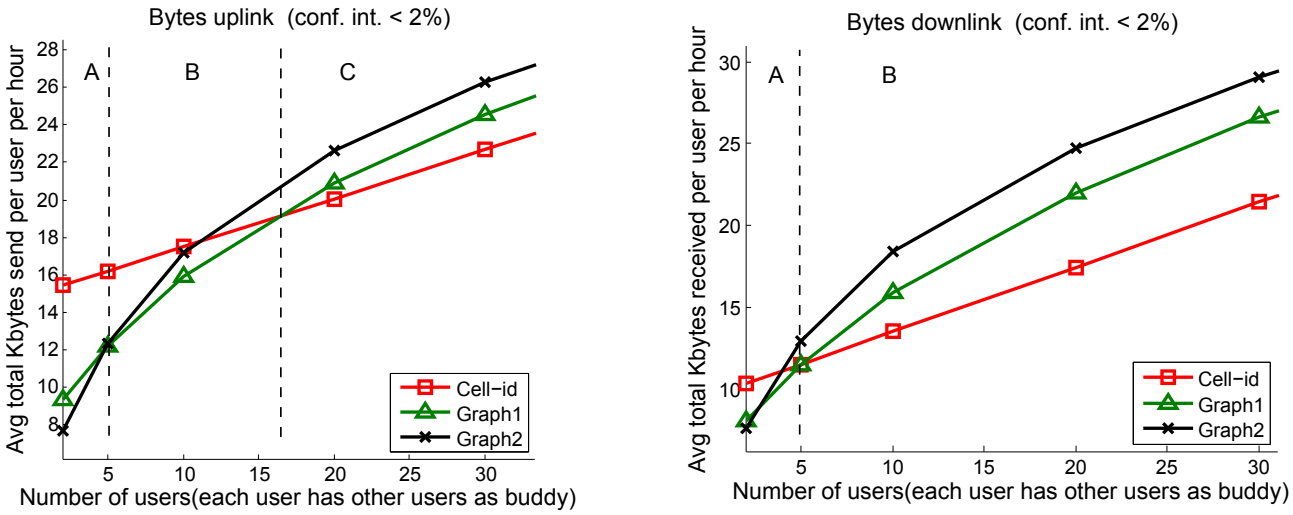
Fig. 5.7: Up and Downlink messages medium area

The output of the simulator is the number of uplink messages (see figure 5.7(a)) and downlink (see figure 5.7(b)) messages. On the Y-axis of the graphs is the average total number of messages per user per hour displayed. The number of users is scaled and each user has the other users as buddy. In figure 5.7(a) the graph algorithms generate less uplink messages, see area A and B. The Graph2 algorithm outperforms Graph1 for a low density of users, see area A. This breakpoint lays around 4 users. The graph algorithms take benefit of the free cells that are available if there is a low density of users.

Around 16 users the number of uplink messages equals for the cell-id algorithm and the Graph1 algorithm. If the downlink messages are also taken into account (total messages per user per hour) this break-even point is around 10 users, for the cell-id and Graph1 algorithm.

The simulator can also keep track of the data usage per user. TCP/IP is simulated over a GPRS connection. In figures 5.8(a) and 5.8(b) the traffic in Kbytes per user per hour is displayed. The same breakpoint as seen with the number of messages returns here. For 5 users or less, the Graph2 algorithm generates the lowest amount of uplink data (area A). Up to 17 users,the Graph1 algorithm generates less uplink data than the cell-id algorithm.

With more users the graph1 algorithm and the graph2 algorithm generate more downlink data. This data is generated by messages initiated from the server. If a user enters another user's free-area, and no proximity is detected, this user has to be informed that his free-area is affected. A new TCP/IP



(a) Uplink traffic

(b) Downlink traffic

Fig. 5.8: Up and Downlink traffic medium area

connection has to be set up with the affected user. This effect can be seen in figure 5.8(b), displaying the downlink traffic. The breakpoint lays around 5 users.

The total traffic in Kbytes per user per hour can be found in figure 5.9. In this medium area of 140 cells the graph2 algorithm only outperform the other algorithms if 3 or less users are in the area, in the figure this is marked with an 'A'. The Graph1 algorithm generates the fewest amount of total data traffic fo 9 users or less, see area 'B'. In area 'C' the cell-id algorithm is performing the best.

So the Graph algorithms need a low density of users to perform well, in terms of number of messages and data traffic. This is due to the fact that the Graph algorithms simply need to have free cells to assign to users. The graph algorithms have the disadvantage of generating more downlink data. If a user enters a free area of another user, the free area of both users needs to be updated, resulting in two messages.

Another simulation run was done on a larger area, because the high density of users in a medium area affects the performance of the graph algorithms. The simulation area contains a cell-layer of 680 cells, this represents an area of 25x25km. In figure 5.10(a) the total data traffic can be found for the simulation run of the large area. As can be concluded from the figure, the Graph2 algorithm performs best up to 10 users (see area 'A' in figure 5.10(a)). In area 'B' the Graph1 algorithm outperforms the other algorithms, in terms

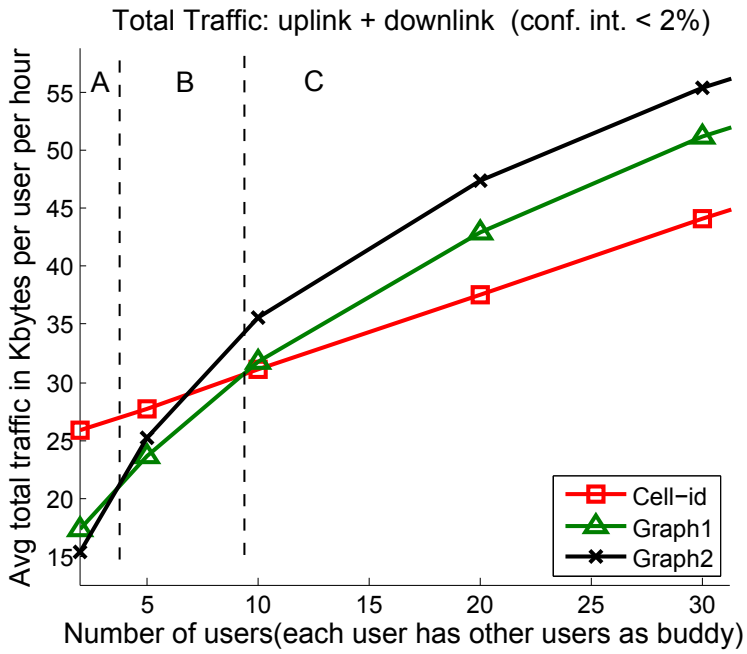


Fig. 5.9: Total traffic medium area

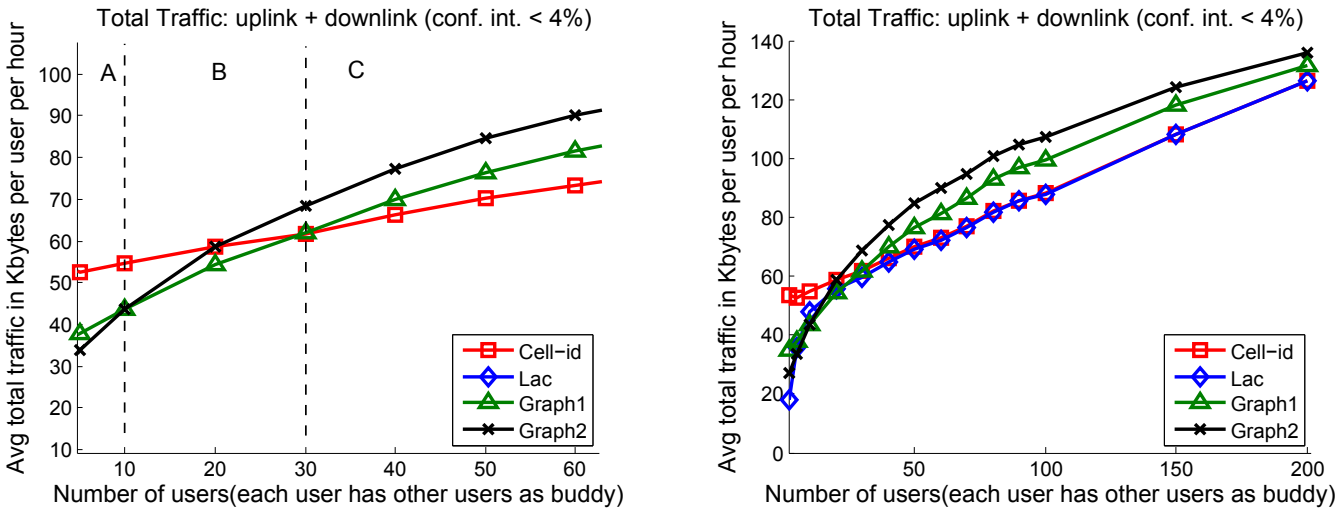
of data traffic. After this breakpoint, 30 users, the basic cell-id generates the lowest amount of data traffic (area 'C').

Compared to the medium area of 140 cells, the Graph algorithms perform better. The LAC algorithm has the advantage of multiple LACs, but the winnings are very small, it is negligible. This is why the LAC algorithm graph is left out of figure 5.10(a).

In figure 5.10(b) it can be seen that, if the density of users is high enough, the algorithms perform the same. In other words, all the algorithms fall back to the cell-id level, as used in the cell-id algorithm. For the Graph algorithms a free-area is simply one cell-id, because there are no free cells available to form a free-area.

5.4 Chasing cars scenario

A typical scenario could be that two users are travelling in the same direction with different speeds. Two users are on the same highway and approach each other several times, because of the other traffic. This special case is simulated to see how the different algorithms react, in terms of messages and data traffic. Two users are simulated using an activity model to generate chasing data. Two nodes are placed on the simulation area, with a distance



(a) Total Traffic large area, zoom

(b) Total Traffic large area

Fig. 5.10: Total data traffic large area (25x25km)

	cell-id	LAC	graph1	graph2
uplink messages	34368	34368	31409	29456
downlink messages	25626	25626	27865	27196
total messages	59994	59994	59274	56652
uplink Kb	1362	1362	1239	1161
downlink Kb	1027	1027	1176	1210
total Kb	2389	2389	2415	2371

Tab. 5.2: Chasing cars scenario results

of 10 km, the medium area cell-id layer is used. The two users chase each other and 1358 proximities are generated, the users speed varies and they will catch up due to different velocities and accelerations. The simulation runtime was 100 hours. The results of the simulation runs, using the chasing cars data, of the four algorithms: cell-id, LAC, Graph1 and Graph2 can be found in table 5.2. This is the average data of 30 runs of 100 hour. The data displayed is the average data per user per 100 hours.

The LAC algorithm has no advantage of using LACs or landcodes in the chasing cars scenario, because the users are in the same LAC all the time. This results in the same messages and the same data traffic as for the cell-id algorithm. The graph2 algorithm can take the advantage of assigning multiple cells to one of the users, only two users are in the area so the

density of users is low. As seen in the distributed scenario simulation the Graph algorithms perform better for a low density of users. So during the chase run it occurs that two or more cells are between the two users. This affects both the total number of messages and the total traffic over 100 hours of chasing data.

5.5 Amount of data used, costs

For the application it is very important that the costs, in terms of money, for the users are acceptable next to the energy consumption. The application has to run continuously to detect proximity with other users. Using the simulation results of the distributed scenario an estimation can be done about the cost per user per month. This scenario simulates a lot of mobility of users and therefore results in a lot of traffic. This is probably the case in rush hour so this is the case for about 3 or 4 hours per work day. On a normal work day the rest of the time you probably be at the office and at night you will be sleeping, depending on the job. Using the distributed scenario the upper bound of costs per month can be calculated.

The data of the medium simulation is taken to illustrate the possible costs for a user with 30 buddies. At 30 buddies the traffic for one user is around 45 Kbytes per hour, see figure 5.9. This results in a data traffic of $45 \times 24 = 1080$Kb, or $\approx$1Mb. Per month this is about 30Mb. This is a lot, but nowadays the providers have flat fee subscriptions. And the distributed scenario is used and as stated before this can be compared to rush hour movement, so in general the data usage will be much lower than 30Mb per month.

5.6 Real user paths

As mentioned in chapter 4 it is also possible to input real user data. A user logs his movement using a GPS and GSM enabled device. An example of two paths of different users can be found in figure 4.2. This path is used to illustrate how the four algorithms react on real measurements. The number of cell-id switches for the short path is 86 and for the other user it is 240. In table 5.3 and 5.4 the results of the algorithm runs can be found.

The LAC algorithm is performing so well, in terms of messages and data traffic, because the two users approach each other from different directions. So they both can use the LACs as free-area. The Graph2 algorithm also performs better than the Graph1 algorithm. Because no other users are in

	cell-id	**LAC**	**graph1**	**graph2**
uplink messages	909	75	549	429
downlink messages	732	50	492	412
total messages	1641	125	1041	841
uplink bytes	36840	3050	22200	17320
downlink bytes	30540	2000	20940	17740
total bytes	67380	5050	42690	35060

Tab. 5.3: Real measurements: user 1

	cell-id	**LAC**	**graph1**	**graph2**
uplink messages	414	30	285	240
downlink messages	380	20	294	264
total messages	794	50	579	504
uplink bytes	16732	1220	11486	9656
downlink bytes	16240	800	12800	11600
total bytes	32972	2020	24286	21256

Tab. 5.4: Real measurements: user 2

the way all the cell-ids are free and can be assigned to the user to form a free-area. The Graph2 algorithm creates a larger free-area than the Graph1 algorithm.

Chapter 6

Conclusions and Future Work

This chapter describes the conclusions found during this research from the simulations. After the conclusions future work is outlined.

6.1 Conclusion

Using the gathered GSM cell data from the Telematics Institute it was found that the average diameter of a cell is 1100m. Cell size varies from 10meter, pico cells, to 13km, macro cells.

Dynamic circles was found the best performing proximity detection algorithm in literature, in terms of number of messages. GPS coordinates are used as location identifier, but GPS needs a line of sight to calculate a position. Inside buildings, next to high buildings and in woods the coverage is poor or even no coverage. GPS cannot be used as continuous location identifier, because of energy consumption.

Two groups of algorithms that detect proximity using the GSM network were defined. The Basic group consists of the algorithms using only the information available from the GSM network. Two algorithms are part of this group, i.e. the cell-id and LAC algorithm. The second group uses a cell-id graph representing the network topology. The Graph1 and Graph2 algorithms are part of this group. All four, client-server deployed, algorithms use the GSM cell-ids as location identifiers for users. The Graph group can offer a proximity distance of multiple cells, the Basic group can only detect proximity in the same cell.

A simulator was developed capable of simulating the four algorithms against realistic generated user movement or real logged paths. The simulator can generate a hexagonal layer or a realistic layer can be used to represent the GSM network topology. The number of messages, the data traffic, the accuracy of the GSM cell layer and the missed proximities or false negatives can be simulated.

For the extrapolated cell layer of the area of Borne the accuracy of the GSM layer is around 1100m. The number of missed proximities for this realistic layer is 3.5% of the average number of proximity detections per user, a static proximity distance of 1100m is used.

To compare the algorithms the proximity distance is defined as the same cell, so if two users are in the same cell proximity is detected. From the simulation of the distributed scenario it can be concluded that for a low density of users the Graph group outperforms the Basic Group. For a large size area, with 680 GSM cells, the Graph1 algorithm outperforms the cell-id algorithm for 30 buddies or less. For the medium size area, with 140 cells, this breakpoint is 17 users. With a very high density of users, each cell is occupied, the algorithms perform the same as the cell-id algorithm. Algorithms using a free-area, as with the Graph group, generate more server initiated messages if a user enters another user's free-area. The LAC algorithm is a minor improvement on the cell-id algorithm, because the size of LACs is large in the Netherlands, i.e. size of city or larger.

The amount of data traffic used by a client with 30 buddies in a month is less than 30Mb.

6.2 Future work

This section describes future work, what could be done to improve the simulations. And what could make the application more valuable.

6.2.1 Adaptive algorithms

The algorithms presented in this thesis are all static algorithms. For future research an adaptive or dynamic algorithm could be defined. In the calculation of the free-area the direction and speed of the user could be taken in to account. Using this information the probability that a user stays longer in it's free-area increases. Also the size of the free-area could be made dependent on the density of users. So if a user is in a city the free-area should be small, if the user is in the country side the free-area should be large.

6.2.2 Realistic cell-id layer

The public cell-id database of Telematica Institute was used during this research to create a realistic layer. Too few measurements resulted in an extrapolated layer used for simulations. For further research a cell-id layer could be better modelled. Solutions to get a better database could be:

- Use the community to gather more cell-id measurements. To really push people to gather data, awards could be given if someone has collected 100 unique cell-ids. This will trigger the user to gather more measurements.
- With a group of people systematically walks patterns to get all the measurements. Repeat this several times to detect possible changes in the topology, due to atmosphere or changes by the operator.
- Sell the application to the operator and use their information about their own topology.

6.2.3 Data traffic

CanuMobiSim and VanetMobiSim were used to generate realistic user movement. These generators lack the ability to generate movement for a user in real-life e.g., go to work, eat, watch TV and go to sleep patterns. If a good estimation of data traffic is needed this type of user movement needs to be generated. It is also possible to give users a GPS enabled mobile GSM device and do a trial. This trial will require a lot of data and users to generate enough proximity detections to get a good estimation of the average data usage in a month.

6.2.4 Multiple operators

For this research only one operator is used to implement proximity detection. It could be that the application is used with different operators, to add more value to the personalized service. If a client has a buddy that uses another operator it is possible to detect proximity. Different operators have different network topologies and different cell sizes. As seen before each operator has a different method to form LACs for example. If the application needs to support multiple operators the following could be done:

- Create a new topology that merges all the different topologies to create a single topology at the server side. This server needs to be in the middle of the two operators and receive location updates of both users. With this merged topology it can detect proximity.
- Make use of a translation database of cell-ids to GPS coordinates. The GPS coordinates are representing the centres of mass of the cell-id the user is in. If this is done for both users with different operators the GPS translated coordinates could be used to detect proximity.

Appendix A

DVD: Simulator and Demos

Efficient proximity detection among mobile clients using the GSM network. ing M.T.Witteman

This DVD is part of the graduation research done on proximity detection using the GSM network. It contains the simulator, all the tools, demo's, other files used to achieve the simulation results that can be found in the Thesis. All parts of the simulator are developed in Matworks Matlab. Each directory contains a Readme.txt explaining the contents of that directory and examples of using the contents.

DVD contents:

AccuracyAndMissedProximities:

This folder contains the simulation done with a realistic layer extrapolated from the area of Borne to get an answer to the subquestion about the accuracy of the GSM network and about the missed proximities.

CanuMobiSim and VanetMobiSim:

These folders contain the work, previously done on generating realistic user movement paths. Both the tools are written in JAVA and are used to generate the data for the simulations. See LargeSizeRun and MediumSizeRun for the usage of these generators.

JAVAwsdlCellLocatorDatabase:

This folder contains the JAVA application that can request cell-ids from the Telematica Database. It can request lat,longs from cell-ids, but it can also request all the cell-ids in a certain latitude longitude rectangle. The

latter is used to get the cell layer from borne.

LargeSizeRun and MediumSizeRun:

These folders contain the batch files to generate the input data en the batch files to scale the number of users per algorithm. These batch files also implement the calculation of the confidence interval on the fly.

ParseLoggedData:

This folder contains the parsers for the log files from the Cumular application. The parsers take the log files and convert it to matlab files, that can be directly imported in Matlab.

ParseNSoutput2matlab:

This folder contains the C++ implementation of the parsers for the JAVA mobility path generators CanuMobiSim and VanetMobiSim. The parsers in matlab format can be found in folder ParsersMobilityData

ParsersMobilityData:

This folder contains the parsers written in matlab to parse the output data of the JAVA mobility path generators CanuMobiSim and VanetMobiSim to matlab ready data

TelinDataViewJAVA and TelinDataViewKML:

These folders contain parsers to view the data from the CellLocator database from Telematica Instituut. The TelinDataViewJAVA views the data retrieved with the JAVA wsdl application found in folder JAVAwsdl-CellLocatorDatabase. The TelinDataViewKML folder contains a viewer that can display the data saved from Google Earth. Johan Koolwaaij and Martin Wibbels have developed a network link that enables you to view the cell-ids in Google Earth, Google Earth has the option to save that data to a KML file. The contents of the file looks a lot like XML. The viewer in TelinDataViewKML use these KML files to display the cells

RealLoggedDataRun:

Simulating Real logged data. The real logged data is used as input for

the four algorithms. The output is the number of messages, up and downlink and the traffic up and downlink.

TraceMap:

Tracemap is a tool written in Matlab to trace real maps and generate a roadmap graph that can be used for CanuMobiSim and VanetMobiSim to define a user graph representing a roadmap.

Demos:

Demos contains several graphical demos of the algorithms and movies of the algorithms

ScaleUser:

A C++ console tool that takes two input files and generates a set of XML input files for CanuMobiSim and VanetMobiSim in which the number of users is scaled.

Other:

Containing other files used for the research

Bibliography

[AEM+04] Arnon Amir, Alon Efrat, Jussi Myllymaki, Lingeshwaran Palaniappan, and Kevin Wampler. Buddy tracking — efficient proximity detection among mobile friends, 2004.

[aw] acbPocketSoft website. acbpocketsoft: acbpowermeter.

[Cha98] Kanwar Chadha. The global positioning system: Challenges in bringing gps to mainstream consumers. *IEEE Explore*, 1998.

[Cho97] Howie Choset. Incremental construction of the generalized voronoi diagram, the generalized voronoi graph, and the hierarchical generalized voronoi graph. In *Proceedings of the First CGC Workshop on Computational Geometry*, October 1997.

[Cit] IBM Citysimulator. alphaworks: City simulator: Overview.

[Dia71] R.B. Dial. A probabilistic multipath traffic assignment model which obviates path enumeration. *Transportation Research*, (5), 1971.

[Dij59] E.W. Dijkstra. *A note on two problems in connection with graphs, Numerische Mathematik*, volume 1. Numerische Mathematik, 1959.

[HF] Jerome Harri and Marco Fiore. Vanetmobisim - vehicular ad hoc network mobility extension to the canumobisim framework.

[HFB05] Jérôme Haerri, Fethi Filali, and Christian Bonnet. On the application of mobility predictions to multipoint relaying in MANETs: kinetic multipoint relays. In *AINTEC 2005, Asian Internet Engineering Conference, December 13 - 15, 2005, Bangkok, Thailand - Also published in LNCS Volume 3837*, Dec 2005.

[HJ04a] Jahan Hassan and Sanjay Jha. Design and analysis of location management schemes for a new light-weight wireless network. *Computer Communications*, 27(8):743–750, 2004.

[HJ04b] Mahbub Hassan and Rai Jain. *High performance TCP/IP networking: Concepts, Issues and Solutions*. Pearson Prentice Hall, Upper Saddle River, NJ 07458, 2004.

[HK03] Shuk Ying Ho and Sai Ho Kwok. The attraction of personalized service for users in mobile commerce: an empirical study. *SIGecom Exch.*, 3(4):10–18, 2003.

[KDK00] S. Kyriazakos, D. Drakoulis, and G. Karetsos. Optimization of the handover algorithm based on the position of the mobile terminals. *IEEE*, 2000.

[KT06] Axel Kupper and Georg Treu. Efficient proximity and separation detection among mobile targets for supporting location-based community services. *SIGMOBILE Mob. Comput. Commun. Rev.*, 10(3):1–12, 2006.

[KW] J. Koolwaaij and M. Wibbels. Gsm cells, mapping cellid to gps coordinates.

[LR05] Pasi Lehtimäki and Kimmo Raivio. A som based approach for visualization of gsm network performance data. In *IEA/AIE'2005: Proceedings of the 18th international conference on Innovations in Applied Artificial Intelligence*, pages 588–598, London, UK, 2005. Springer-Verlag.

[MRB07] E. Meeuwissen, W. Romijn, and J. Brok. Estimated time of arrival and estimated location, 2007.

[NR04] Subrata Nandi and Manish K. Raushan. An efficient implementation of distance-based update scheme using directional cell identification codes. In *IWDC*, page 535, 2004.

[osc05] Validation of an improved location-based handover algorithm using gsm measurement data. *IEEE Transactions on Mobile Computing*, 4(5):530–536, 2005. Member-Hsin-Piao Lin and Student Member-Rong-Terng Juang and Member-Ding-Bing Lin.

[Sch03] Jochen Schiller. *Mobile communications 2nd.* Addison-Wesley Pearson Education, Edinburgh Gate, Harlow, England, 2003.

[S.M87] S.M.Ross. *Introduction to Probability and Statistics for Engineers and Scientists*. John Wiley and Sons, New York, 1987.

[Stea] Charles Steinfield. The development of location based services in mobile commerce. http://www.msu.edu/user/steinfie/elifelbschap.pdf.

[Steb] Illya Stephanov. Canumobisim - a framework for user mobility modeling.

[TWK06] Georg Treu, Thomas Wilder, and Axel Kupper. Efficient proximity detection among mobile targets with dead reckoning. In *MobiWac '06: Proceedings of the international workshop on Mobility management and wireless access*, pages 75–83, New York, NY, USA, 2006. ACM Press.

[UE] USA and Europe. Everything you want to know about e911 and e112. http://www.globallocate.com/RESOURCES/RESOURCES_MAIN_f3.htm.

[weba] HTC website. Htc: Products: Htc p3600.

[webb] Mathworks Matlab website. Mathworks matlab.