Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

MARIA FLEUREN[1], HINNERK STÜBEN,
GIDEON ZEGWAARD[2]

MoDySim
A parallel dynamical UMTS simulator

[1]TNO Telecom, St. Paulusstraat 4, 2264 XZ Leidschendam, The Netherlands
[2]QQQ Delft, Kluyverweg 2a, 2629 HT Delft, The Netherlands

Talk presented by H. Stüben at *ParCo 2003* — Parallel Computing 2003, 2–5 September 2003, Dresden, Germany

ZIB-Report 03–26 (September 2003)

MoDySim — A parallel dynamical UMTS simulator*

M.J. Fleuren[a], H. Stüben[b] and G.F. Zegwaard[c]

[a]TNO Telecom, St. Paulusstraat 4, 2264 XZ Leidschendam, The Netherlands

[b]Konrad-Zuse-Zentrum für Informationstechnik Berlin (ZIB), Takustr. 7, 14195 Berlin, Germany

[c]QQQ Delft, Kluyverweg 2a, 2629 HT Delft, The Netherlands

UMTS is a 3rd generation mobile telecommunication system which enables multi-service and multi-bit rate communication going beyond the possibilities of previous systems. The simulator MoDySim models UMTS in great detail. Characteristics of UMTS such as soft hand-over and the interdependency of load and capacity among neighbouring cells are challenges for the parallelisation of such a system. In this paper we explain how the software was parallelised and present performance results of a UMTS simulation for the city of Berlin.

1. INTRODUCTION

UMTS is a 3rd generation mobile communication system. After GSM (2nd generation), which supported mainly voice calls, GPRS (2.5th generation) opened the way for packet switched data communication. UMTS will enable true multi-service and multi-bit rate communication. Services such as video telephony or file downloading will become possible on mobile phones and laptops.

UMTS differs in many respects from GSM and GPRS. At the eve of implementation and deployment of the system still many aspects need to be investigated. MoDySim is a programme that simulates UMTS in great detail in order to study the behaviour of large UMTS systems realistically.

MoDySim was developed from scratch in the EU 5th Framework project MOMENTUM. The aim of MOMENTUM was to develop network planning algorithms for UMTS in order to obtain optimised networks. It is important to be able to check the outcome of the planning process with a detailed simulator like MoDySim.

MoDySim was implemented in an interdisciplinary effort by the MOMENTUM partners TNO Telecom (formerly KPN Research), QQQ Delft, and ZIB. The three partners combined UMTS network knowledge, professional programming, and experience in parallelisation.

MoDySim is a dynamical simulator (in contrast to a snap-shot simulator), i.e. it simulates the behaviour of the system over time taking into account movements and usage behaviour of the users as well as the dynamics of the network control. The sizes of the systems to be investigated (cities) make it necessary to parallelise the programme in order to shorten its run time to a reasonable length. This paper focuses on the parallelisation aspects of MoDySim. The main problems to be dealt with in the parallelisation are:

- The interference in UMTS is a long range effect. Interference couples large parts of the system, while one would like to exploit data locality of small sub-systems in a parallel programme.
- The effects of cell breathing and soft hand-over imply an absence of a natural (unique) home process for the mobile stations.
- Power control implies (in principle) extensive communication between the processes (in a real UMTS system powers of mobile and base stations are updated every 0.66 ms).

2. INTRODUCTION TO UMTS

This section provides some background for understanding the dynamics of UMTS and the resulting problems of parallelisation.

*Talk presented by H. Stüben at *ParCo 2003* — Parallel Computing 2003, 2–5 September 2003, Dresden, Germany.

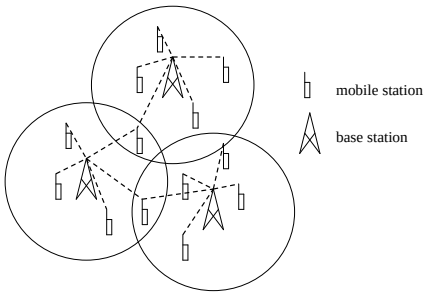

Figure 1. Typical layout of a UMTS network. The dashed lines indicate connections between the base station and the mobile station.

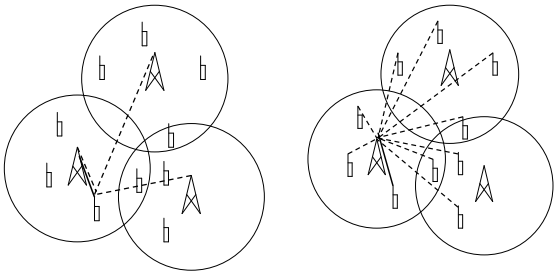

Figure 2. Interference. In the downlink signals of base stations interfere at the mobile stations (left). In the uplink signals of mobile sations interfere at the base stations (right).

In UMTS base stations (BS) are located to receive signals from mobile stations (MS) (see Figure 1) and transport it (via fixed networks) to either servers or to other BSs which can then transmit the signal to another MS.

Unlike GSM and GPRS, UMTS uses the same two frequencies in the uplink (MS → BS) and downlink (BS → MS) across all cells of a network (operators may ultimately use two such paired frequencies). Different data streams are multiplexed. A signal can only be properly detected if the signal to interference ratio (C/I) at the receiver is above a certain target value. This means that the UMTS system is *interference* limited.

As interference is caused by all transmitters (BSs in the downlink, MSs in the uplink, see Figure 2), the available capacity in a cell is directly related to the concurrent transmissions in its surrounding cells. This implies that the capacity in a cell is not a fixed value, but depends on both the interference situation of its surrounding cells and in its own cell. For example, the cells surrounding a heavily loaded cell will typically have less capacity available in that situation. In UMTS the interference couples the cells, and therefore couples the capacity in the cells.

An important dynamical effect in UMTS is *cell breathing:* heavily loaded cells will shrink, while lowly loaded ones will grow (see Figure 3). UMTS can handle services with different bit rates and delay requirements: voice, video conferencing, web browsing, e-mail retrieval, etc. In UMTS higher bit rates require higher power. This implies that the cell breathing dynamics is not only influenced by moving users but also by the type of service.

UMTS has a *power control* mechanism making sure that the transmission power is such that the required C/I at the receiver is exactly met. Every 0.66 ms powers are adapted. A higher power would pollute the system with extra interference, and unnecessarily use precious capacity.

Another technical characteristic of UMTS is *soft hand-over.* When an MS moves, its connection to the BS may become weaker and even become lost. The MS is then handed over to a near BS (not necessarily the nearest) with a stronger signal reception. In UMTS the MS can be simultaneously connected to several (typically two to three) BSs. This situation is called soft hand-over. This means that in the downlink, the MS combines the receiving signals, and in the uplink that the signals received by the BSs are combined higher up in the network.

Two other terms will be mentioned in the text: *call admission control* and *congestion control.* When a new user wants to set up a call, the MS has to ask permission to the BS to connect. If there is not enough capacity available, the request will be rejected in order to protect the quality of the present users. This is called call admission control.

Congestion control starts when the interference becomes too high (e.g. if too many calls have been admitted or too many users move to the same BS). It will block all new calls, reduce the power of the transmitters, and, as the last resort, terminate calls in order to reduce interference.

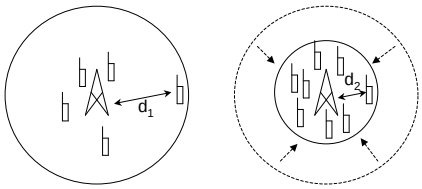

Figure 3. Illustration of cell breathing. Small cells can carry a higher load than large cells.

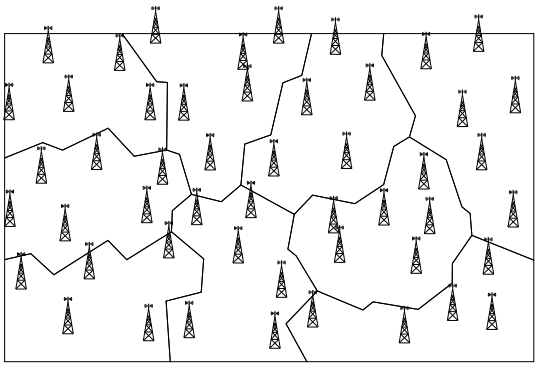

Figure 4. Example of a domain decomposition. Each domain contains the active base stations (see text) treated by one process.

3. OVERVIEW OF MoDySim

3.1. Software design goals

The goal was to develop a dynamical UMTS system simulator with a high level of detail and fast enough to handle large scenarios, e.g. a city with at the order of hundred BSs with realistic traffic load and a mix of services. The system should incorporate realistic user behaviour (movement and usage of services), realistic power control, soft hand-over, call admission control, and congestion control.

3.2. Choice of basic algorithm

A fundamental decision at the beginning of the project was to choose between event-driven and time-driven simulation. Event-driven algorithms are popular for simulating mobile communication networks. However, these kind of algorithms do not seem well suited for a (parallel) UMTS simulator. Events have a stochastic nature. There are stochastic events in UMTS but in addition there is the regular heartbeat of the power control signals. This regular heartbeat is the shortest time-scale in the simulation.

Events have to be processed in the right order. As a consequence the whole (event-driven) system could only proceed in the rhythm of the heartbeat. This observation makes the time-driven simulation a natural choice.

In the time-driven framework the simulation resembles a classical molecular dynamics simulation. Concerning the parallelisation one is therefore naturally led to a domain decomposition approach. The approach we have taken is explained in section 4.

3.3. Main loop

The body of the main loop of MoDySim represents one heartbeat. It looks like this (the abbreviations MS and BS indicate which groups of stations are involved):

(1) MS: Update locations (includes arrivals of new calls and clean-up of finished mobiles)
(2) MS, BS: Calculate quality of signals
(3) MS: Soft hand-over requests
(4) BS: Call admission control
(5) MS: Update activities
(6) MS, BS: Power update (includes interference calculation)
(7) BS: Congestion dropping
(8) MS: Quality dropping

Some more detailed explanations are in order. In step (1) the mobiles move to their next location. The mobility of the users is realistic (e.g. along roads) and with varying speeds. The calculation of signals in (2) is based on propagation grids that describe signal loss. The propagation grids model

real environments (e.g. buildings, streets, mountains, rivers). Based on the new values of signals the mobiles decide on sending hand-over requests to base stations in (3). In step (4) the base stations decide upon all requests. In addition to the hand-over requests (3) there are also requests to admit new calls from step (1).

Update of call activities (5) results in adjustments of bit rates depending on the service (e.g. voice call, picture download). Bit rates affect required powers. The higher the bit rate the higher the power required.

The power update (6) comprises the update of total base station powers, sending of power control signals to mobiles, adjustment of mobile powers (mobiles in soft hand-over have to consider all received control signals) and interference of mobile signals at the base stations and vice versa. Finally in (7) and (8) dropping of calls may occur due to base station congestion or low signal quality at mobiles (the aim is, of course, to minimise both).

4. PARALLELISATION

4.1. Basic approach

The basic approach to parallelise MoDySim is a domain decomposition as indicated in Figure 4. Domains usually overlap and one has to make sure that overlapping areas are being updating correctly with data from remote processes. In a UMTS simulator the overlap areas contain base stations and mobiles.

While it is clear how to assign home processes to base stations this is not so obvious for the mobiles. In a system with *hard* hand-over every mobile is connected to one base station (at most). Therefore, the mobile's home process would be the corresponding base station's home process. In UMTS there is *soft* hand-over implying that a mobile can be connected to two (in practice up to three) base stations, possibly sitting on two (or three) processes. Typically 30–40% of the mobiles are in soft hand-over (the value depends on the parameter settings of the operator).

In order to reduce bookkeeping and to minimise communication between processes there are no overlap areas in MoDySim. Special concepts were instead introduced for the base stations and mobiles.

4.2. Concept for mobiles

Mobiles that are not in soft hand-over exist only on the home process of the serving base station. When a mobile is in soft hand-over, a copy of the mobile exists on *every* home process of the base stations it is linked to. The copies of the mobiles have to behave identical. In other words the copies of the mobiles perform identical operations, the results of which would have to be communicated otherwise. The consequence is that no mobile data has to be exchanged between processes (except for the action of copying a mobile).

4.3. Concept for base stations

Similar to the introduction of copies of mobiles we introduced copies of base stations. In contrast to the copies of mobiles the functionality of the base station copies is reduced. Actually, all base stations reside on all processes. We differentiate between *active* and *shadow* base stations. Active base stations can perform all tasks while the shadow base stations can only perform some tasks, especially call admission control (they can receive requests and make decisions). The active base stations are the ones one would think of in terms of the domain decomposition, the shadow base stations are there in addition. All base stations participate in the interference calculation which is a global operation.

The construction is such that every mobile (including its copies) can always contact any base station without exchanging data between processes. An active base station can contact all the mobiles linked to it while a shadow base station cannot.

The result (or aim) of this construction is that there are basically only two communication patterns. There are (a) global sums and (b) point-to-point date exchange between neighbouring processes. In both cases only base station data is communicated.

4.4. Changes to the sequential version

In this section the changes to the sequential version are described for each step of the main loop introduced in section 3.3.

A general change is that loops over base stations become loops over active, shadow, or all base stations. Loops over mobiles do not have to be changed.

Steps (2), (3), (5), and (8) of the main loop can be left unchanged.

Step (1). No changes are needed for updating the mobiles' locations. New calls (mobiles) are brought into existence on the process hosting the geometrically nearest base station.

Step (2). No changes are necessary. However, it should be mentioned that in this part we did not decompose the so called environment grids, which are input to the calculation of signals. As a consequence the computer memory needed does not decrease when using more processes. The reason for not decomposing theses grids is that it is then guaranteed that the interference calculation can be done without approximation. In practice this was no restriction. The Berlin scenario fitted into 2 GByte of memory per process.

Step (4). Call admission control turned out to be quite subtle in the parallel version. This is the place where mobiles are copied or moved between processes. One has to make sure that the objects representing the mobiles are identical (especially all copies have to know how many copies exist, because the number is needed in the interference calculation). Call admission decisions are made locally. This is possible because only a few parameters are needed at the base station (those parameters had been broadcasted directly after they were updated).

Step (6). Updating the powers requires global sums. One global sum is necessary in the interference calculation. All base stations sum up the contributions of the mobiles that are on their process (each contribution is divided by the number of copies of the corresponding mobile in order to obtain the correct contribution from each mobile). Then the global sum is performed. Only the active base stations can calculate new powers. These have to be distributed to the shadow base stations. For this all-to-all communication the global sum routine is used, too.

Step (7). Only the active base stations can determine the congestion status and drop mobiles. On each process a list of dropped mobiles is set up and these lists are broadcasted.

4.5. Other issues

Random numbers. We have used the Luxury Random Number Generator (RANLUX) [1]. Every mobile and every active base station generates random numbers individually. By proper initialisation it is guaranteed that the individual sequences of random numbers are uncorrelated.

Load balancing. Load balancing in MoDySim is static. It was assumed that the researcher setting up a simulation scenario has external means to devise a decomposition that leads to acceptably balanced load. This turned out to be true in the cases that were run up to now.

Implementation. MoDySim was written in C++. A fundamental decision for the parallelisation was whether to use OpenMP or MPI. We have decided to use MPI. The main reasons are platform independence and availability. Another big advantage is that the parallel code can be much better modularised. OpenMP requires basically that all loops have to be touched. In MPI parallelisation practically means to add a few calls to data exchange routines and the implementation of a communication class. Once this class is written the code can be further developed without knowledge of MPI.

Testing. We have tested the parallel version by demanding that the output does not depend on the decomposition. Due to rounding errors in the global sum the output cannot be bit-wise identical. In principle it could happen that different rounding leads to a different yes/no decision. In our tests such "butterfly effects" did not occur. Results agreed at the precision at which values are logged (6 decimal places).

5. SCALING

The parallel code was tested and run on an IBM p690 computer. The computer has shared memory nodes with 8 and 32 CPUs respectively and 2 GByte of memory per CPU.

In the first parallel runs of MoDySim an artificial test scenario was used. The scenario consisted of base stations located at the centers of a hexagonal grid with 7 cells. For this scenario a speed-up of

about 5 was obtained on 7 processes. At the time of writing the first realistic scenario was run. The city of Berlin was the first case studied. The Berlin scenario consists of 74 sectorised base stations covering the inner city districts. Table 1 shows the scaling of execution times for this scenario.

Table 1
Scaling of execution times for a UMTS simulation of the city of Berlin.

processes	execution time	speed-up
1	71 h	1.0
7	17 h	4.2
14	18 h	3.9

6. RELATED WORK

Writing a parallel dynamical UMTS simulator like MoDySim is pioneering work. A related project is SEACORN [2] which also develops a parallel UMTS simulator. However, we have found no paper about their parallelisation work. Traditionally, mobile communication simulations are event-driven, while MoDySim is time-driven. A generic event-driven dynamical simulator is WiNeS [3]. A recent work on parallelising event-driven simulations is [4].

7. DISCUSSION AND CONCLUSION

In the first realistic simulation MoDySim achieves a useful parallel speed-up. Although, the programme only scales to a moderate number of processes. On 7 processes we observe a speed-up of 4. On 14 processes the programme runs slightly longer than on 7 processes. The main cause for this slowing down are the global sums that are needed in the interference calculation. On the 7 processes the global sums need 5.5% of the execution time while on 14 processes the sums need 42% of the time.

One design goal of MoDySim was to keep the programme as simple as possible. A consequence was that no cut-off was introduced in the interference calculation. With cut-off there would be no global operations which should result in higher scalability. In addition the memory per process would scale (now the memory requirements per process is independent of the number of processes, see section 4.4). Another simplification was to stay with static load balancing.

It is interesting to compare our scaling result with those obtained in [4]. The scaling reported there is 4.2–5.8 out of 8 which is similar to our result. One has to keep in mind that in [4] event-driven simulations were studied. A technical implication is that for parallel event-driven simulations shared memory computers are preferred. Because MoDySim was parallelised with MPI it runs also on distributed memory architectures, especially PC-clusters.

8. ACKNOWLEDGEMENTS

This work was supported by the European Commission under contract number IST-2000-28088. Computations were performed on the IBM p690 system of Norddeutscher Verbund für Hoch- und Höchstleistungsrechnen (HLRN).

REFERENCES

[1] M. Lüscher, Comput. Phys. Commun. 79 (1994) 100.
F. James, Comput. Phys. Commun. 79 (1994) 111.
[2] `http://seacorn.ptinovacao.pt`
[3] J. Deissner, G. Fettweis, J. Fischer, D. Hunold, R. Lehnert, M. Schweigel, A. Steil, J. Voigt and J. Wagner, A Development Platform for the Design and Optimization of Mobile Radio Networks, in: D. Baum, N. Müller, R. Rösler (eds.), Kurzvorträge der 10. GI/ITG-Fachtagung Messung, Modellierung und Bewertung von Rechen- und Kommunikationssystemen MMB'99, Universität Trier, Mathematik/Informatik, Forschungsbericht Nr. 99–17, pp. 129–133.
[4] M. Liljenstam, Parallel Simulation of Radio Resource Management in Wireless Cellular Networks, Dissertation, Royal Institute of Technology, Stockholm, Sweden, 2000.