Traffic Simulation and the Location of Mobile Units in Wireless Communication Systems[1]

Mauro Nacif Rocha[2]
Departamento de Informática
Universidade Federal de Viçosa – MG – Brazil
e-mail: nacif@dcc.ufmg.br

Geraldo Robson Mateus and Soraia Lúcia da Silva
Departamento de Ciência da Computação
Universidade Federal de Minas Gerais – Brazil
e-mail: [mateus,soraia]@dcc.ufmg.br

ABSTRACT

This paper presents some different aspects and problems related to the process of building a simulator for tracking mobile units in wireless communication systems. The work also considers several approaches related to city area models, location area planning and the storage of user profiles, which are needed for the simulations.

INTRODUCTION

The design of wireless communication systems is usually based on a cellular architecture, which consists on a group of fixed Base Stations (BS) covering the service area and interconnected by a fixed backbone network. The coverage area of each BS is called a cell. In addition to the fixed BS we have a number of Mobile Units (MUs) that can freely roam throughout the entire service area.

In such systems, MUs place and receive calls through a wireless medium. Although placing a call may not seem very difficult, receiving calls implies in much more effort for the management system. Units may be moving anywhere in a large geographic area, following many mobility patterns, so finding them may not be straightforward.

In analog cellular systems like AMPS, when a unit receives a call, the system sends a paging signal hoping that the MU will send a response signal and accept the connection. This paging signal is usually sent throughout a large or maybe the entire geographic area covered by the system. This so called *blanket paging*, although efficient in terms of finding the MU, wastes precious resources, as we often have only one radio channel to perform this paging.

In modern digital cellular systems like GSM, TDMA and CDMA, large geographic areas are divided into *Location Areas* (LAs) or *Paging Areas* (PAs), or both. Each LA has some kind of signature that is constantly monitored by each MU. Whenever an MU crosses an LA boundary it sends

---

[1] This work has been supported by CAPES, Telemig and Project SIAM/DCC/UFMG, grant MCT/FINEP/PRONEX number 76.97.1016.00
[2] Currently a PH.D. level student at "Universidade Federal de Minas Gerais" – Brazil.

a *Location Update* (LU) signal, which is detected by the nearest cell. The system can then track the MU and, whenever it receives a call, page only the LA in which he last updated his position. This allows the system to seek several MUs at a time, each of them in a separate LA.

The problem with this approach is that highly mobile units tend to waste a lot of power (usually from low capacity batteries) sending LU signals. This drastically lowers the lifetime of the batteries and clotters the medium with radio signals, which may not be used. Many papers have been written proposing algorithms which basically transfer great part of the effort of finding the MU to the supporting system. These methods include statistical prediction, individual user patterns, dynamic change of LAs or PAs, hierarchical LAs among others.

In any case, the research and test of these algorithms rely entirely upon some kind of simulator, which needs to be flexible enough so that many different approaches can be easily fed in and fast enough to track hundreds of thousands (even millions) of MUs. Our goal is to present a simulator built to aid on our studies of different algorithms for location management. Since MUs could as well be cellular telephones, machines, portable computers or even programs and data, this could be of great help to anyone working in any wireless environment.

1. A CITY AREA MODEL

Although it would be nice to use a real city model like the one in Figure 1a, this would impose on high processing demands, which we may not have, and limit the number of MUs we would be able to track.

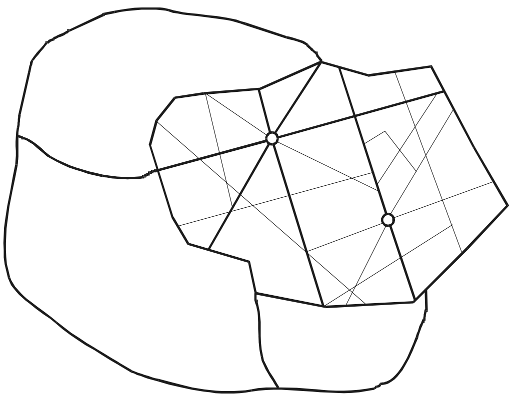

Figure 1a

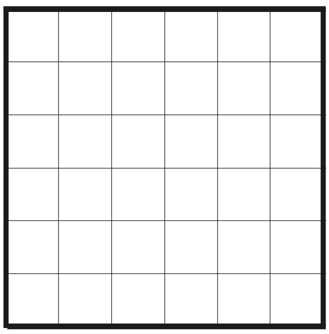

Figure 1b

On the other side, Figure 1b may look a bit artificial, but it is extremely easy to code. Called the Manhattan Model or Grid Model [6,13], it perfectly serves our needs, and because of its simplicity, allows us to track hundreds of thousands of MUs. Another city area model which is used in [11] can be seen below (Fig. 2).

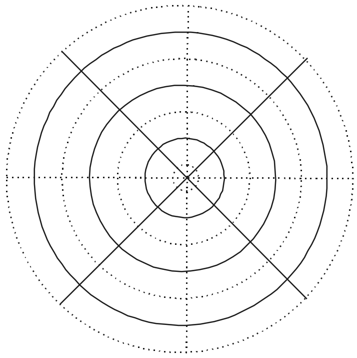

Figure 2.

Dotted lines indicate area borders and solid lines are radial and peripheral highways. Note that as we only need to track mobile units and test the behavior of our location strategies as well as the LA size and distribution, any reasonable city area model can do well.

In our simulator, we used a square grid with 12.8 Km sides. The streets are spaced every 128 meters, and are numbered 0, 1, 2,...*Ns*, with odd numbered streets going west or north, even numbered streets going east or south and the most outer streets allowing traffic on both directions. Speed of the MUs is controlled independently for the inner and the outer streets, typical values being 30 and 50 Km/h respectively (Fig. 3).

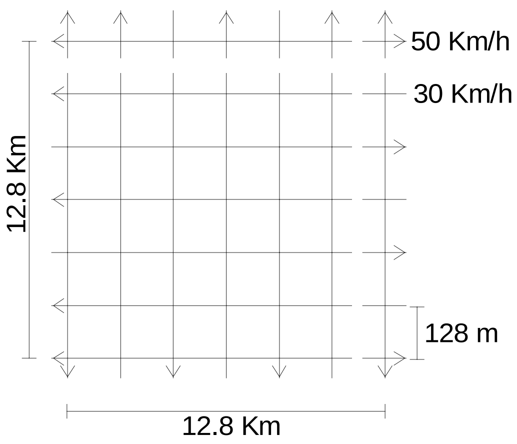

Figure 3.

2. REGIONS OF INTEREST

We can also define several areas within the city which we call *regions of interest*. Figure 4 shows the window where we can define such areas simply clicking on the colors and dragging inside the little square blocks. The program then stores this information for later use. Each square block corresponds to a 256 x 256m area, which is 4 city blocks in our model.

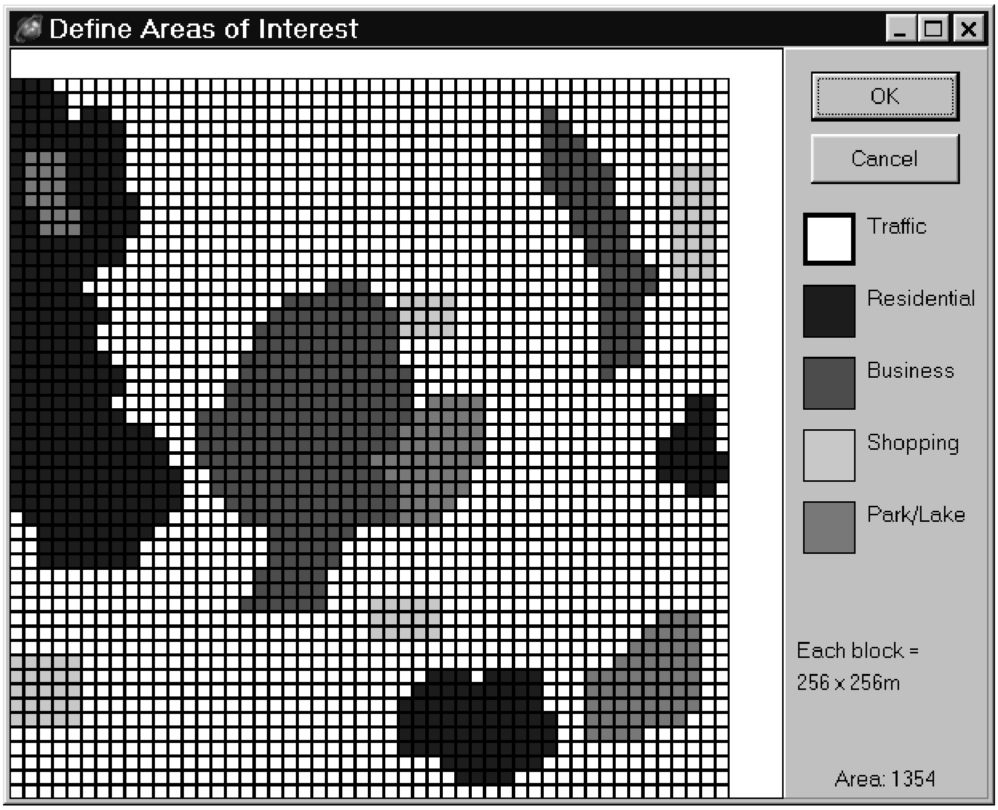

Figure 4 – A visual editor for defining regions of interest.

The main reason for this is that MUs are more likely to move among these areas, and most of them move from home to work and then back home again. If we could statically or even dynamically gather profiles of each MU, we could use this information together with the above regions of interest for locating an MU whenever we need so, or whenever his location falls beyond his last LU signal.

3. USER PROFILES

User profiles could be gathered statically (e.g. a form filled upon registration) and updated dynamically by the system to reflect changes in a unit's behavior. Basic information would include the following:

- Home address
- Work address
- Profession (this could indicate a degree of mobility)

We could also build timetables to schedule the MU's activities. Figure 5 shows an example for a timetable built for an ordinary worker. It tells that, for example, from 7:30h to noon the MU is certainly at its workplace, and from noon to 13:30h he will be found at "Lunch" with probability $p = 0.75$, at a Bank with $p = 0.20$, or at a Shopping Center with $p = 0.05$. This would be used only for simulation purposes, not for the location algorithms.

Start	End	Location	Local Probabilities					
0:00h	7:00h	Home	2%	Party				
7:00h	7:30h	Traffic						
7:30h	12:00h	Work						
12:00h	13:30h	Lunch	20%	Bank	5%	Shopping		
13:30h	18:00h	Work						
18:00h	18:30h	Traffic						
18:00h	24:00h	Home	30%	Shopping	10%	Restaurant	5%	Visit

Figure 5 – Timetable built for an ordinary worker.

Highly mobile units, like delivery boys and taxies, could have random walks around the whole city (with higher probabilities between home and work areas), and housekeepers could have much simpler timetables than the one above. To reduce memory constraints, MUs can be divided in categories, like the three just mentioned, and we would have only one table for each category.

4. LOCATION UPDATE (LU) SIGNALS

For our introduction and first analysis purposes, we fed our simulator with three different approaches on LU signals: following the IS-41 standard, by time and by distance.

IS-41

This protocol is used in most of today's digital cellular systems, including TDMA and D-AMPS. It states that whenever an MU crosses an LA boundary, it sends an LU signal. The draw-backs of this approach have already been discussed in the introduction. We then introduced a new parameter, called *Tolerance*, which is given as the amount of seconds that the MU should wait do send his LU signal after it crosses an LA boundary.

This tolerance could help reduce the unnecessary LU signals sent to the system while the MU is walking along an LA boundary. We could also have this parameter given in distance instead of time. If further tests show similar results between time and distance tolerances, obviously track-ing time would be much easier to implement in any MU.

Preliminary results show that although this additional parameter does reduce the amount of paging done by the MU, it also makes it harder for the system to exactly find his position in any given time, so we would have to implement better location strategies.

By Time

This simple approach states that, at each given time interval, say every 30 seconds, the MU sends an LU signal, regardless if he has crossed any LA boundary or not. This is the approach used by analog cellular systems, like AMPS. For low mobile MUs, this isn't very good because it wastes battery. For highly mobile MUs this approach could lengthen battery life, but we could have a hard time locating it in digital systems.

By Distance

This approach states that every time the distance from the point that the MU's last LU signal was sent and his actual position equals a certain distance threshold parameter, than the MU sends an LU signal (Fig. 6b). This distance traversed by the MU could be calculated with the aid of a GPS positioning system. But then we would have new problems, such as the space occupied, the cost of the GPS, and more important, the energy it would consume, which is exactly what we want to reduce.

By Movements

In this approach, every time the sum of all straight line moves equal a certain distance threshold parameter, than the MU sends an LU signal. This can be seen in Fig. 6a. Studies show [6,7] that this approach is generally worst than by time or by distance, so we will drop it altogether.

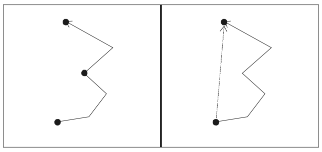

Figure 6 – LU by movements (a) and by distance (b). Black dots indicate LU signal positions.

5. LOCATING MOBILE UNITS

So far we have only implemented a strait and simple algorithm which looks for the MU only inside the LA where he sent its last LU signal, and nowhere else. While this works fine for plain IS-41, giving us a 100% hit rate (we will not consider MU and system overhead delay problems which could arise with fast moving units), it is not appropriate for LU algorithms that try to save MU energy resources.

6. LOCATION AREA PLANNING

By now we are considering fixed partially overlapping circular cells, each one with a 200m radius. LAs are also fixed, as shown in Fig. 7. Although this is fairly easy to implement in the simulator, it would be better to arrange the cells in LAs so that we could have smaller LAs in densely populated areas and larger LAs otherwise.

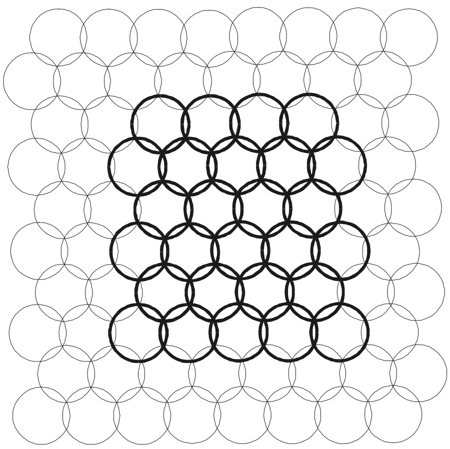

Figure 7 – bold cells indicate a new Location Area

Although we can implement models for obtaining optimal distribution for LAs [18], the great number of parameters involved generally lead to the using of heuristics [12,18]. But even then we always come up with static LAs.

We would also like to test the behavior of dynamic LAs. In this case, LAs could be resized depending on the daytime period (e.g. larger LAs from midnight to 6 AM), as well as the profile of each mobile unit (e.g. considering his degree of mobility and known speed – Fig. 8). This possibility of having LAs being defined dynamically for each MU every time he sends a new LU signal can be seen in [12,14]. Another possibility is to group LAs, building hierarchical models which could be used statically for all MUs or modified dynamically depending on the MU's profiles [16].

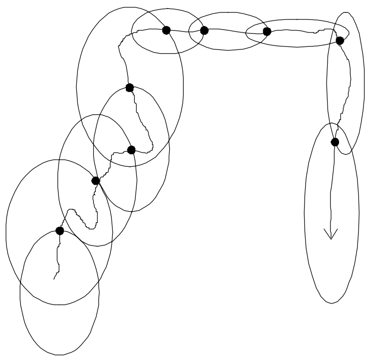

Figure 8 – Illustration of location area recalculation with time. In this example, the MU might first be driving around a city and then enter a high-speed route.

Our purpose is to study new and better resource saving algorithms that, together with old and new LU approaches, user profiles and LA planning, would be economically and technically feasible to implement in third and forth generation systems, which include MUs, Base Stations and underlying systems.

7. SIMULATING

A pseudocode for the main routine of the simulation program can be seen in figure 9. Time increments are done in 1-second units. Calls arrive in average every 15 min. for each unit, with mean duration of 2 min. We used uniform distribution for both. The user simply chooses the total time of simulation and the total number of MUs that will be created. We have been able to test up to 1,200,000 MUs on a 128 MB Pentium II. The initial time t_0 is pre-defined as being 6:00 AM. Other options as discussed above can also be chosen prior to simulation start.

```
CreateCells;
CreateUsers;
TimeElapsed := 0;
t := t0;
repeat
   for i := 0 to NumMobilUnits-1 do
      with MobilUnit[i] do begin
         Move;
         if ReadyForCall then begin
            if FindMobilUnit then begin
               inc( Hits );
               StartCall;
               end
            else inc( Misses );
         end;
         inc( TimeElapsed );
         inc( t );
      end;
until (TimeElapsed >= TotalTime);
DestroyUsers;
DestroyCells;
```

Figure 9 - Pseudocode for the main routine of the simulation program.

Besides controlling MU movements, including when to start, where to go and which route to follow, the simulator also has to control other aspects such as:

- Whenever each MU leaves one LA and enters another (for IS-41).
- Whenever each MU receives a call.
- Trying to locate an MU whenever it receives a call.

The measure of our efficiency lies in the number of locate misses done by the system and the number of LU signals sent by the units. We need to minimize both, which is conflicting, because lowering one almost always implies in raising the other. Besides, we also need to keep an eye on some practical aspects. For instance, we can obtain very smart algorithms that, along with very few LU signals sent by the unit, can efficiently track him. But this same system should be able to track hundreds of units per second, which can make the algorithm infeasible if it doesn't respond quickly enough.

We also need to concern the aspects of data retreival. Although the system would need to interact with a Distributed Data Base (DDB) for retrieving and updating system-related and specially unit-related data (Home Location Register – HLR / Visitor Location Register –VLR), we can presume that this DDB management is very efficient, with negligible response times. This could work for some algorithms, but others could impose on impracticably heavy data retrieving and update demands, no matter which DDB Manager (DDBM) we choose.

8. RESULTS

Our first tests, done over an 8 hour simulated period, show obvious results, but that nonetheless will serve as a first comparison basis and as references for future tests. We used 1000 MUs, each of them with a mobility degree of 50% (they move approximately 50% of the time). We are not yet considering areas of interest and user profiles, so their destinations are calculated randomly. We also assume negligible overhead for HLR/VLR updates and no processing delay. Figure 10 shows these results.

LU Method	**Parameter**	**Hit Rate**	**LUs / Unit / hour**
IS-41	Tolerance = 0s	100%	12,5
IS-41	Tolerance = 2s	99,7%	12,4
IS-41	Tolerance = 4s	99,0%	12,3
by Time	Time = 288s	57,6%	12,5
by Time	Time = 5s	99,4%	720
by Distance	Dist. = 880m	55,6%	12,5
by Distance	Dist. = 18m	99,4%	607

Figure 10 **–** Results of an 8 hour simulated period for several LU methods.

We can observe that, for the Time and Distance approaches to obtain the hit rate of IS-41, we need to send an enormous amount of LU signals. This shows that, to use these methods, we must consider an intelligent algorithm for unit locating.

We also tested simulator time responses to see how many units we could track at a reasonable time, and to see how the program scales with simulation period and number of MUs. The results below (Fig. 11) were taken from a Pentium II/233 with 128MB of RAM running Windows NT 4.0.

Simulation Period (hours)	**# of MUs**	**Time ([hh:]mm:ss)**		
		IS-41	**Time**	**Distance**
3	10,000	11:27	4:09	4:17
6	10,000	22:58	8:19	8:27
3	50,000	57:38	21:05	21:42
6	50,000	1:53:47	42:48	43:17

Figure 11 **–** Simulator response times.

CONCLUSIONS

We showed several different aspects and problems related to building a simulator for tracking mobile units in wireless communication systems. We also considered several approaches on city area models, location area planning and the storage of user profiles needed for the simulations, as well as considerations upon algorithm and DDBM efficiency.

Although our simulator seems to have good response times, it would be nice to increase its basic performance to track a larger number of MUs and use more sophisticated algorithms for locating these units.

Besides tracking MUs, it would also be desirable to have control of calling traffic and density, database traffic (with both signaling and data traffic), as well as handoff control. This could help on LA design strategies and channel allocation, both in static and dynamic approaches. Although not being specific for this purpose, data traffic and user density in both conventional and wireless networks could then be simulated and studied, using this simulator as a more general purpose tool.

REFERENCES

[1] A. Abutaleb, V.O.K. Li, "Location Update Optimization in Personal Communication Systems," *Wireless Networks*, Vol. 3, pp. 205-216, 1997.

[2] A. Abutaleb, V.O.K. Li, "Paging Strategy Optimization in Personal Communication Systems," *Wireless Networks*, Vol. 3, pp. 195-204, 1997.

[3] I.F. Akyildiz, J.S.M. Ho, "A Mobile User Location Update and Paging Mechanism Under Delay Contraints," *Proc. of the ACM SIGCOMM'95*, Cambridge, Mass., pp. 244-255, 1995.

[4] I.F. Akyildiz, J.S.M. Ho, "Dynamic Mobile User Location Update for Wireless PCS Networks," *Wireless Networks*, Vol. 1, pp. 187-196, 1995.

[5] I.F. Akyildiz, J.S.M. Ho, "On Location Management for Personal Communications Networks," *IEEE Communications Magazine*, pp. 138-145, Sep. 1996.

[6] A. Bar-Noy, I. Kessler, "Tracking Mobile Users in Wireless Networks," *IEEE Trans. on Information Theory*, Vol. 39, pp. 1877-1886, Nov. 1993.

[7] A. Bar-Noy, I. Kessler, M. Sidi, "Mobile Users: To update or not to Update?," *Wireless Networks Journal*, Vol. 1, No. 2, pp. 175-186, 1995.

[8] A. Bar-Noy, I. Kessler, M. Naghshineh, "Topology-Based Tracking Strategies for Personal Communication Networks," *Baltzer Journals*, Feb. 1996.

[9] P.A. Fishwick, "Computer Simulation: The Art and Science of Digital World Construction," University of Florida, Computer & Information Science and Engineering Department, T.R.95-017, Sep. 1995.

[10] Y.B. Lin, S.Y. Hwang, "Comparing the PCS Location Tracking Stratagies," *IEEE Trans. on Vehicular Technology*, Vol. 45, No. 1, pp. 114-121, Feb. 1996.

[11] G.L. Lyberopoulos, J.G. Markoulidakis, D.V. Polymeros, D.F. Tsirkas, E.D. Sykas, "Intelligent Paging Strategies for Third Generation Mobile Telecommunications Systems," *IEEE Transactions on Vehicular Technology*, Vol. 44, No. 3, pp. 543-554, Aug. 1995.

[12] J.G. Markoulidakis, G.L. Lyberopoulos, D.F. Tsirkas, E.D. Sykas, "Evaluation of Location Area Planning Scenarios in Future Mobile Telecommunication Systems," *Wireless Networks*, Vol. 1, pp. 17-29, 1995.

[13] M. Middendorf, “Manhattan Channel Routing is NP-Complete Under Truly Restricted Settings,” *Chicago Journal of Theoretical Computer Science*, Vol. 2, pp. 1-19, 1996.

[14] C. Rose, R. Yates, “Location Uncertainty in Mobile Networks: a theoretical framework,” *IEEE Communications Magazine*, Vol. 32, No. 2, Feb. 1997.

[15] J. Short, R. Bagrodia, L. Kleinrock, “Mobile Wireless Network System Simulation,” *Wireless Networks Journal*, Vol. 1, No. 4, 1995.

[16] S. Tabbane, “Location Management Methods for Third-Generation Mobile Systems,” *IEEE Communications Magazine*, pp. 72-84, Aug. 1997.

[17] G.R. Mateus, A.A.F. Loureiro, *Introdução à Computação Móvel*, 11ª Escola de Computação, DCC/IM, COPPE/Sistemas, NCE/UFRJ, 1998.

[18] P.R.L. Gondim, “Gerência de Mobilidade em Redes de Comunicação Móvel Celular,” Ph.D. Thesis in Electrical Engineering, PUC-RJ, 1998.