Mobile Information Systems 5 (2009) 255–289
DOI 10.3233/MIS-2009-0084
IOS Press

Accurate mobility modeling and location prediction based on pattern analysis of handover series in mobile networks

Péter Fülöp, Sándor Imre, Sándor Szabó and Tamás Szálka
Budapest University of Technology and Economics, Department of Telecommunication, 2, Magyar tudósok körútja, Budapest 1117, Hungary
E-mail: {fulopp,imre,szabos}@hit.bme.hu; szalkat@mul.hu

Abstract. The efficient dimensioning of cellular wireless access networks depends highly on the accuracy of the underlying mathematical models of user distribution and traffic estimations. Mobility prediction also considered as an effective method contributing to the accuracy of IP multicast based multimedia transmissions, and ad hoc routing algorithms. In this paper we focus on the tradeoff between the accuracy and the complexity of the mathematical models used to describe user movements in the network. We propose mobility model extension, in order to utilize user's movement history thus providing more accurate results than other widely used models in the literature. The new models are applicable in real-life scenarios, because these rely on additional information effectively available in cellular networks (e.g. handover history), too. The complexity of the proposed models is analyzed, and the accuracy is justified by means of simulation.

Keywords: Location prediction, mobility model, Markovian approach, random walk, complexity, accuracy, handover, user movements

1. Introduction

Random Walk Mobility model is often used in network planning and in analyzing network algorithms, because of its simplicity [1]. On the other hand this very simple model presumes unrealistic conditions, like uniform user distribution in the mobile network. However, in real-life networks, geographical characteristics, such as streets and parks influence the cell residence time (dwell time) and movement directions of users in the network, and result in a non-uniform user density. While these models are appropriate for mathematical analysis, easy to use in simulations and for trace-generation, they fail to capture important characteristics of mobility patterns in specific environments, e.g. time variance, location dependence, unique speed and dwell-time distributions, etc. [3,6,12,24].

User movements in a cellular network can be described as a time-series of radio cells the user visited. The handover event of active connections (e.g. cell boundary crossing) is recorded in the network management system's logs, thus the information can be extracted from the management system of cellular mobile networks, such as GSM/GPRS/UMTS networks. The users movements are described by the dwell-time and outgoing probabilities (the probability of a user leaving for each neighboring cell). These parameters can be calculated for each cell based on the time-series of visited cells of the users. However, in some cases, these two parameters – dwell-time and outgoing probabilities – are not enough

to capture all the information in the time-series of user movements. In many situations, the outgoing probabilities are correlated with the incoming direction of the users, so the movement contains memory.

This paper is organized as follows. In Chapter 2 we give a brief on the mobility models, and in Chapter 3 we summarize their applications in mobility prediction, Call Admission Control, etc. [22].

In Chapter 4 we introduce extensions to widely used mobility models in order to enhance the accuracy of the models, meanwhile aiming at keeping the models as simple, as possible. In Chapter 5 we propose a novel mobility model, called Path Based Mobility model, especially designed to predict movements in cellular networks in urban areas. We calculate the theoretical accuracy of the models, and compare them by simulation to other models can be found in the literature in Chapter 6. Chapter 7 concludes the paper.

The results of this paper are applicable in engineering tasks of location based services, network dimensioning, can provide input for more effective Call Admission Control algorithms in order to ensure user's satisfaction and optimal resource usage in cellular wireless mobile networks [19].

2. Mobility models in the literature

Different mobility models have been proposed in the literature to cope with user mobility in different wireless and mobile networks (e.g. cellular networks, ad hoc networks etc.) In this chapter we give a short overview on the most widely used mobility models.

In the *Random Walk* mobility model [15], the node moves from its current location to a new location by randomly choosing a direction and a speed. The Random Walk model defines user movement from one position to the next with memoryless, randomly selected speed and direction. Each movement in the Random Walk Mobility Model occurs in either a constant time interval t or a constant distance traveled d. When a mobile node reaches a simulation boundary, it simply bounces off the simulation border with an angle determined by the incoming direction, then the node continues along this new path. Many derivatives of the Random Walk Mobility Model have been developed including the *one*, *two*, *three* – and d-dimensional walks.

Chiang's mobility model [21] defines the *Probabilistic Version of Random Walk* model, which utilizes a probability matrix to determine the position of a particular mobile network in the next time step. The position of the node is represented by three different states for position x and three different states for position y. In this mobility model state 0 represents the current (x or y) position of a given node, state 1 represents the node's previous (x or y) position, and state 2 represents the node's next position if the node continues to move in the same direction. The probability matrix used in the probabilistic random walk model is

$$P = \begin{bmatrix} P(0,0) & P(0,1) & P(0,2) \\ P(1,0) & P(1,1) & P(1,2) \\ P(2,0) & P(2,1) & P(2,2) \end{bmatrix},$$

where each entry $P(a,b)$ represents the probability that a node will go from state a to state b. The values within this matrix are used for updates to both the node's x and y position.

In the *Random Waypoint* model [7] the user stays at a particular location for a specified time period before moving on to the next in a randomly chosen direction with speed uniformly distributed between zero and maximum speed. The model derived from the Random Walk model breaks down the entire movement of the user into a series of pause and motion periods.

A density wave is the clustering of nodes in one part of the simulation area. The *Random Direction Mobility Model* [9] was created to overcome *density waves* in the average number of neighbors produced

by the Random Waypoint Mobility Model. In case of the Random Waypoint Mobility Model, this clustering occurs near the center of the simulation area.

The *Modified Random Direction Mobility Model* is a slight modification to the Random Direction Mobility Model [9]. In this modified version, the node continues to choose random directions, but they are no longer forced to travel to the simulation boundary before stopping to change direction. Instead, a node chooses a random direction and selects a destination anywhere along that direction of travel. The node then pauses at this destination before choosing a new random direction.

In the *Boundless Simulation Area Mobility Model*, a velocity vector $\vec{v} = (v, \theta)$ is used to describe a node's velocity v as well as its direction θ. In this model a relationship between the previous direction of travel and velocity of a node with its current direction of travel and velocity exists [27]. Both the velocity vector and the position are updated at every Δt time steps according to the following formulas:

$$v(t + \Delta t) = \min[\max(v(t) + \Delta v, 0), v_{\max}],$$
$$\theta(t + \Delta t) = \theta(t) + \Delta\theta$$
$$x(t + \Delta t) = x(t) + v(t) + \cos\theta(t),$$
$$y(t + \Delta t) = y(t) + v(t) + \sin\theta(t).$$

Here, $V_{\max}$ is the maximum velocity, Δv is the change in velocity which is uniformly distributed between $[-A_{\max}\Delta t, A_{\max}At]$, and $A_{\max}$ is the maximum acceleration of a given node. The parameter $\Delta\theta$ means the change in direction which is uniformly distributed between $[-\alpha\Delta t, \alpha\Delta t]$], and α is the maximum angular change in the direction.

To adapt different levels of randomness, one can use the *Gauss-Markov Mobility Model.* In this model initially each node is assigned a current speed and direction. In this mobility model at fixed intervals of time, n movement occurs by updating the speed and direction of each node. The value of speed and direction at the *n-th* instance are calculated based upon the value of speed and direction at the (n+1)-*st* instance and a random variable using the following equations:

$$s_n = \alpha s_{n-1} + (1 - \alpha)\,\vec{s} + \sqrt{(1 - \alpha^2)}s_{x_{n-1}},$$
$$d_n = \alpha d_{n-1} + (1 - \alpha)\,\vec{d} + \sqrt{(1 - \alpha^2)}d_{x_{n-1}},$$

where $\vec{s}$ and $\vec{d}$ are the new speed and direction of the node at time interval n, and $\alpha = 0$ is the tuning parameter used to vary the randomness, while s and d are constants representing the mean value of speed and direction. The parameters $s_{x_{n-1}}$ and $d_{x_{n-1}}$ are random variables from a Gaussian distribution. Totally random values are obtained by setting $\alpha = 0$ and linear motion is obtained by setting $\alpha = 1$ [5], while intermediate levels of randomness are obtained by varying the value of α between 0 and 1.

In the *City Section Mobility Model* we represent a section of a city where the ad hoc network exists [21]. The streets and speed limits on the streets are based on the type of city being simulated. The streets might form a grid in the downtown area of the city with a high-speed highway near the border of the simulation area to represent a loop around the city.

If a flexible mobility framework for hybrid motion patterns is needed, one can rely on the *Mobility Vector model* [26]. A mobility vector expresses the mobility of a node as the sum of two sub vectors: the Base Vector $\vec{B} = (bx_v, by_v)$ and the Deviation vector $\vec{V} = (vx_v, vy_v)$. The base vector defines the major direction and speed of the node while the deviation vector stores the mobility deviation from the base vector. The mobility vector $\vec{M}$ is expressed as $\vec{M} = B + \alpha V$ where α is an acceleration factor.

In mobile networks there are many situations where it is necessary to model the behavior of nodes as they move together as a group. These models are called group mobility models.

The most general of these group models is the *Reference Point Group Mobility* (*RPGM*) model. In the Reference Point Group Mobility model [25], the motion of the group center completely characterizes the movement of its corresponding group of nodes. The RPGM model represents the random motion of a group of nodes as well as the random motion of each individual node within the group. The logical center for the group is used to calculate group motion via a group motion vector $\vec{GM}$. In this mobility model the individual nodes randomly move about their own pre-defined reference points, whose movements depend on the group movement. When the updated reference points $RP\,(t+1)$ are calculated, they are combined with a random motion vector $\vec{RM}$, to represent the random motion of each node about its individual reference point. Specifically, three group mobility models (Column, Nomadic, and Pursue) can be implemented as special cases of the RPGM model.

In the *Exponential Correlated Random* Mobility Model [25] a motion function is used to create node movements. Given a node or group position at time t, and $\vec{b}\,(t)$ is used to define the next position at time $(t+1)$:

$$\vec{b}\,(t+1) = \vec{b}\,(t)\,e^{-\frac{1}{\tau}} + \left(\sigma\sqrt{1-\left(e^{-\frac{1}{\tau}}\right)^2}\right) r,$$

where τ adjusts the rate of change from the node's previous location to its new location, thus small τ equates to large change and r is a random Gaussian variable with variance σ. We notice, that it is not easy to create a given motion pattern by selecting appropriate values for (r, σ) in the Exponential Correlated Random Mobility Model [25].

This *Column Mobility Model* [20] represents a set of nodes that move around a given line or column, which is moving in a forward direction. The model is useful for scanning or searching purposes and a slight modification of the Column Mobility Model allows the individual nodes to follow one another. Each node is placed in relation to its reference point in the reference grid; the node is then allowed to move randomly around its reference point via an entity mobility model.

In the *Nomadic Community Mobility Model*, each node uses an entity mobility model like the Random Walk Mobility Model, to roam around a given reference point. The model represents groups of nodes that collectively move from one point to another [14]. However, within each community or groups, the individuals move in random ways. When the reference point changes, all nodes in the group travel to the new area defined by the reference point and then begin roaming around the new reference point.

The *Pursue Mobility Model* [14] represents nodes tracking a particular target. The model consists of a single update equation for the new position of each node. The *random vector* value is obtained via an entity mobility model like the Random Walk Mobility Model. The amount of randomness is limited in order to maintain effective tracking of the node being pursued. The model combines the current position of a node, a random vector, and an acceleration function to calculate the next position of the node.

3. Related work

Mobility prediction provides useful input for dimensioning and planning of wireless mobile networks, ad hoc routing algorithms, efficient multicast transmission and call admission control [19,23].

In this chapter we present a brief summary on mobility prediction algorithms.

The *shadow cluster* scheme [8] estimates future resource requirements in a collection of cells in which a mobile is likely to visit in the future. The shadow cluster model makes its prediction based on the mobile's previous routes. In this model, the highway traffic with various constant speeds is simulated and users travel in forward and backward directions. The shadow cluster model improves estimation of resources and decision of call admission. In the study by Chao and Chen (1997), user mobility is estimated based on the aggregate history of handoff observed in each cell Shadow Cluster Concept takes its prediction, based on the mobile's previous routes.

The *proximity model* [2] minimizes the requirement for precise mobility information and computes the initial baseline link availability assuming random independent mobility. This model aims to quantify the future proximity of adjacent nodes and reflects the future stability of a given link. The model adapts future computations depending on the expected time-to-failure of the link. The total link availability between two nodes m and n is expressed as:

$$A^T_{m,n}(t) = A^i_{m,n}(t) P_i + A^c_{m,n}(t)(1 - P_i),$$

where $A^T_{m,n}(t)$ is total link availability, $A^i_{m,n}(t)$ is the availability when mobility is independent and $A^c_{m,n}(t)$ is the availability when mobility is correlated. The metric reflects independent or correlated behavior as given by the value of P_i. A value of $P_i = 1$ reflects independent movement with respect to the total link availability.

The *Sectorized ad hoc mobility prediction* scheme [18] achieves maximum accuracy in movement prediction. In this model the prediction process should be restricted to areas of high cluster change probability. The sectorized ad hoc mobility prediction scheme makes use of the cluster-sector numbering scheme to predict user movements in an ad hoc network.

In the *cluster based* [18] model the cluster head has complete knowledge of each of its member nodes. In this model the location of the user is defined with respect to its position with that of the cluster head. Assuming a circular cluster structure there is a region of the cluster in which all the nodes belonging to the cluster are in closest proximity to each other. All nodes in this region of the cluster are within communication range of each other, and the nodes in this region of the cluster will not satisfy the requirements for membership to any of the neighbouring clusters.

A prediction mechanism for *link expiration time* between any two ad hoc nodes has been observed [11] to enhance various unicast and multicast ad hoc routing protocols. In this model, if node i and node j at positions (x_i, y_i) and (x_j, y_j) are travelling at speeds v_i and v_j with moving directions θ_i and θ_j respectively with a transmission range r, then the time period D_t during which they would stay connected is predicted as:

$$D_t = \frac{-(ab + cd) + \sqrt{(a^2 + c^2) r^2 - (ad + bc)^2}}{a^2 + c^2},$$

where $a = v_i \cos\theta_i - v_j \cos\theta_j, b = x_i - x_j, c = v_i \sin\theta_i - v_j \sin\theta_j$, and $d = y_i - y_j$.

In wireless ad hoc networks, *network partitioning* occurs when mobile nodes moving with diverse mobility patterns. If the network consists of two mobility groups C_j and C_k, and each moving at velocities W_j, and W_k, the relative mobility is obtained by fixing one group stationary. The effective velocity W_{jk} at which C_k is moving away from C_j is given as $W_{jk} = W_k + (-W_j)$. Assuming that all groups have a circular coverage region of diameter D wherein the nodes are uniformly distributed and

are in overlap, C_k must move past a distance of the diameter D of C_j's coverage area. The time taken for the two groups to change from total overlap to complete separation is given as:

$$T_{jk} = \frac{D}{\sqrt{w_{jk,x}^2 + w_{jk,y}^2}},$$

where $W_{jk} = \left(w_{jk,x}^{+} w_{jk,y}\right)$. The partition prediction method employed in a clustering algorithm exhibits perfect accuracy of node classification [13].

4. Methods to enhance the accuracy of mobility models

The basic idea of our work is to utilize the additional mutual information available in the time-series of mobile users' movement patterns in cellular mobile networks. In our work we assume that given traces can be extracted from the mobile service provider's network history. The network management dataset consists of network management signals that were transferred in the network during the examined time interval. Beside many other network parameters and properties, two main information sets can be recovered:

- the cell-path that each user visited before
- the time intervals users have spent in each cell

The series of visited cells is crucial to analyze the similarities in the users' motion. Based on the motion patterns of the terminals amongst the cells, we can describe some drifts of the users' motion in a given cell or point. A drift may be caused by geographical or infrastructural objects (like highways, etc.) or some time-dependent circumstances (like mass events, concert, football matches, etc.).

From a mathematical point of view, the drift of the motion can be interpreted as different transition probabilities from one cell to another. A probability-vector can be defined for each cell that describes the probabilities of moving from the source cell to an adjacent one. This vector is called Handover Vector (HOV) [10]. The HOV is a discrete distribution, it contains transition probabilities to neighboring cells on the condition that the user moves out from the given cell [4]. The HOV might have time dependent components (e.g. the morning or the afternoon rush hours). We describe the usage of the HOV in Section 4.2.

The cell dwell time is an important parameter that helps to estimate the velocity of the motion. Velocity can vary in a wide range since the users frequently change their speed or motion direction.

Motion dynamics depends on the speed and the drift. From the network operator's point of view the accurate prediction of these components are useful to estimate the users' position in the near future, especially in case of handoffs. The analytically precise model that is able to predict both, helps the network operator in dynamic resource planning for short intervals based on actual network state. A long-term resource planning scheme (i.e. day-night scheme) can be supported by a short-term prediction that can follow dynamic escalations of call-initiations or degradation of the quality of the radio channel.

In this section we propose extensions to well known mobility models, in order to model motion drift and velocity with direction and cell dwell time prediction.

4.1. Random walk

Although the Random Walk model is easy to be used, it has disadvantages in user motion modeling:

– Without any improvements, the RW model is not capable of simulating the mobility in an environment where geographical or infrastructural objects determine the motion behavior. Beside this, the RW model simulates the user distribution in the network in a uniform way which is clearly not applicable in real-life situations.
– The model steps in each time slot. The previously visited states or the origin state are recurrent in one and two dimensions, but the model does not allow the same state in two following time slots. Thus the time spent in each state is not taken into account, each time tick means a transition to another cell that can be interpreted as a constant velocity user-motion.
– The model does not include the user's motion history, the states that were visited in the past. The uniformly distributed successor directions in a given state are not precise enough in a real-life application. It can be seen that the possibility of moving forward in a user drift is higher than stepping backward. The sophisticated weighted possibilities of successor directions can be constructed by knowing the previously visited cells [7] or by mapping the geographical or infrastructural circumstances of the area covered by the given cell.

4.2. Random walk model extensions

We have decided to use the RW model since it can be applied in case of a wide range of motion patterns (from pedestrian walk to highways) and can be specialized for uncommon scenarios also. The focus of our work is the elimination of the drawbacks outlined in Section 4.1. With a few extensions the RW model becomes able to more accurately estimate future user distribution in a network or cell cluster.

Since the RW model assumes a motion with constant absolute value of velocity it cannot simulate different movement speeds. The best method to implement this feature is ensuring the possibility of staying in the same RW state for arbitrary amount of time. This can be achieved by two different approaches that mostly give similar results.

According to the basic RW model the user remains in the actual cell until the end of the actual time slot. The lengths of slots are equal, since it is a discrete time system. We propose two methods to replace the constant time slot-lengths with a distribution that converges better to real-life motion patterns.

The handoff trace can be used to calculate several dwell times for each cell. Based on these derived data-series, we modeled each cell by a phase-type (PH) system, which produces a distribution of the cell dwell time with the appropriate absorption time. The phase-type system is defined in Eq. (1) with a transient matrix (**A**), a vector that contains the transition intensities to the absorbing state (**a**) and the initializing vector (α).

$$Q = \begin{bmatrix} \mathbf{A} & \mathbf{a} \\ 0 & 0 \end{bmatrix} \quad pdf_{PH} = f(A, \alpha) \tag{1}$$

The Cell Phase Type System is defined based on the trace, a unique distribution can be created for each cell. It is started when a user registers in the given cell, and the model unregisters the user when the PH system absorbs. Immediately after the absorption the user registers into one of the adjacent cells.

In Fig. 1 the simplest PH system can be seen, combined with the uniform successor distribution. The PH contains one transient state which means that the absorption time is exponentially distributed.

Based on the handoff trace another dwell time simulation method can be derived. We provided the possibility of staying in the same cell for the next fixed length time slot as an extension of the RW model. There is an additional transition, the loopback direction. The simple RW model does not allow the user to stay in the same state, so our proposition enables the model to simulate different cell dwell times.

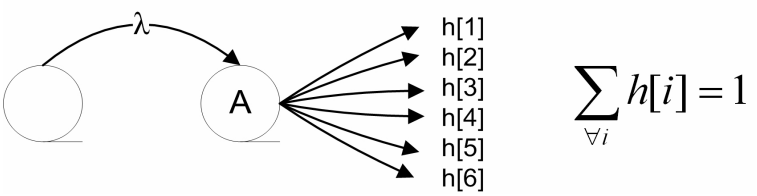


Fig. 1. Exponentially distributed (exp(λ)) dwell time simulator with HOV.

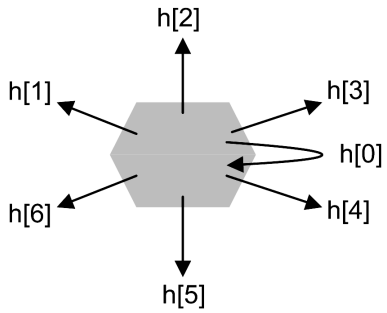


Fig. 2. Looping transition in the modified RW structure.

Formally that means the probability of stepping into the same cell at the end of the slot, such as a looping step shown in Fig. 2.

With the correct tuning of the loopback value, arbitrary distributions of cell dwell times can be simulated if the time slot length is appropriately chosen.

$$\begin{aligned} \mathbf{h}' &= \left[\mathbf{h}'[0], \mathbf{h}'[1], \mathbf{h}'[2], \mathbf{h}'[3], \mathbf{h}'[4], \mathbf{h}'[5], \mathbf{h}'[6]\right] \\ \mathbf{h}'[0] &= pstay \\ \mathbf{h}'[i] &= (1 - pstay) \cdot \mathbf{h}[i] \end{aligned} \tag{2}$$

Equation (2) shows that the outgoing transition probabilities have to be weighted in order to keep them as a distribution. This model is the Extended Random Walk (ExtRW).

The remaining drawback of the Extended Random Walk model is the lack of handling different motion directions. Our extension substitutes the uniformly distributed random successor direction in ExtRW to a special distribution that is valid only in the given cell. This distribution is represented by the Handover Vector (HOV). The HOV can be determined from the trace of handoffs made in an arbitrary network. The given data of the trace provides measures of the relative frequency of handoffs between each adjacent cell-pairs. Based on the relative frequency, the probabilities of the HOV can be easily calculated. The HOV is cell-specific, each cell has its own geographical properties that affects the users' motion drift and the transition probabilities into the adjacent cells.

The Extended Random Walk model combined with the cell-dependent transition probabilities gives the best estimation of future user distribution. In following sections the simulation results prove that the Handover Vector can simulate the constant or time-dependent changing density of the users distributed in the different cells of the network.

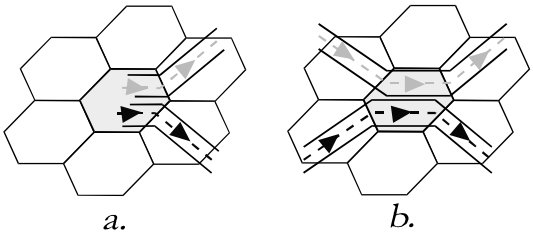


Fig. 3. User prediction methods; a. model without memory, unknown where is the users come from; b. model with memory, the previous steps of the users taken in account.

4.3. The effect of memory in the model

The Random Walk model, as it was analyzed in the previous section, is the most commonly used mobility model to describe individual movement behavior. The extended RW and HOV eliminate the disadvantages of the RW model with variable dwell time and weighted transition probabilities calculated from known parameters of the system in the past.

The limit of accuracy even in the most sophisticated HOV model is caused by the lack of memory, since the RW-like models do not possess the knowledge of the actual distribution of recent moments. Beside the trends of user movement flows of a longer period in the past, the application of the recent user locations have a capital importance in variable, directional user motion. Neglecting the actual transition series of a user in the cluster, i.e. the model is memoryless, the estimation works with a significantly higher error rate. Figure 3 shows a simple example which represents the effect of memory in the model. If we consider two roads shown is Fig. 3b, the accuracy of transition probability estimations are better when the model knows where the users come from than the RW-like estimation which cannot differentiate the users on the two roads (Fig. 3a).

To clarify the error rate of a memoryless HOV model compared to an algorithm that possesses memory, we use the example cells shown in Fig. 3. Let us define the following parameters:

– the number of incoming users on the upper road at timeslot t is $in1_t$,
– the number of incoming users on the lower road at timeslot t is $in2_t$,
– similarly the number of users leaving on the upper road at timeslot t is $out1_t$,
– the number of users leaving on the lower road at timeslot t is $out2_t$,
– and the user movement directions with a simple transition matrix

$$\mathbf{P} = \begin{bmatrix} p_{11}\, p_{12} \\ p_{21}\, p_{22} \end{bmatrix}$$

The HOV model in Fig. 3a calculates the number of leaving users (*out1*, *out2*) with a historical estimation of the **P** matrix, and with the sum of *in1* and *in2* (the total number of users in the observed cell), but without the knowledge of *in1*, *in2*.

In our proposed algorithm, the use of *in1*, *in2* means that the model calculates based on the information of the incoming users. The incoming users can be considered uniform or different (marked users). Since the latter is more accurate, in this comparison we use marked users.

Assume that the historical estimation of the HOV model's **P** matrix is based on the previous timeslot. That is the P($out1_{t+1}$) and P($out2_{t+1}$) probabilities are $out1_t$/($out1_t$+$out2_t$) and $out2_t$/($out1_t$+$out2_t$), respectively. Applying the same assumption on the algorithm with memory, the number of leaving users can be calculated with the **P** matrix itself, that is $out1_{t+1} = in1_t$*p_{11}+ $in2_t$*p_{21} and $out2_{t+1} = in1_t$*p_{12}+

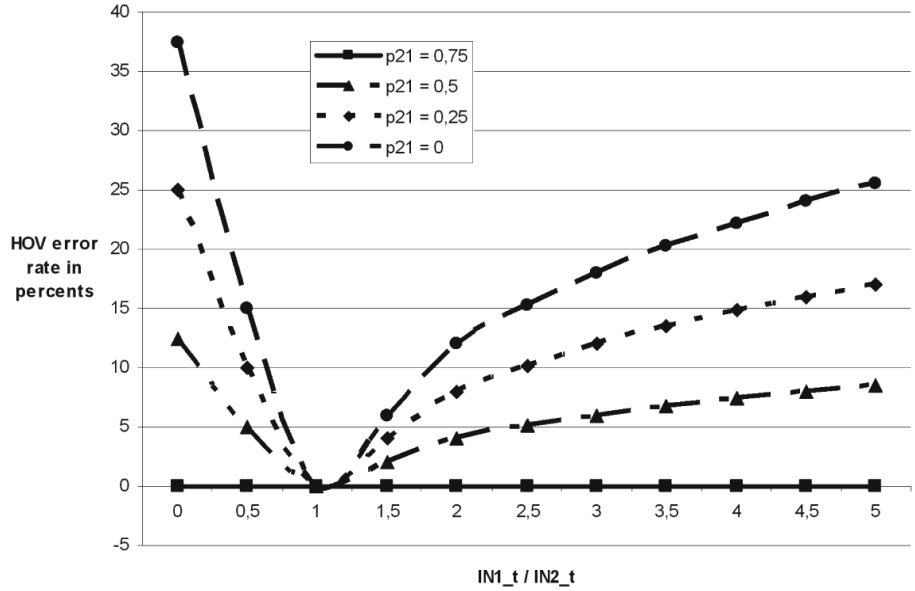


Fig. 4. HOV prediction error in percents – Given $in1_{t-1}/in2_{t-1} = 1$ and P = $\{\{0.75, 0.25\}.\{p_{21}, 1\text{-}p_{21}\}\}$, p_{21} plotted with four different values.

$in2_t * p_{22}$. At a given and constant **P** matrix let us assume that the incoming user distribution varies, that is the $in1_t/in2_t$ ratio (Incoming Distribution – ID) changes. Figure 4. shows the error of HOV compared the estimation using memory.

The HOV model works with error if ID_{t-1} is different from ID_t which is caused by the fact that the HOV historical **P**-estimation in this special case equals the number of leaving users of the previous timeslot. That is it does not include the actual ID_t value. Contrarily the memory-model calculates with the actual number of incoming users and the **P** matrix itself, which results the exact probabilities of the leaving users distribution. The error rate caused by the lack of memory increases as the variance of ID increases, that is $in1_t/in2_t$ ratio changes.

Using memory cannot enhance furthermore the accuracy of the estimation if $p_{21} = p_{11}$ and Fig. 4. shows a constant zero error rate ($p_{11} = p_{21} = 0.75$). In this case the leaving direction of each user is independent of the incoming direction and the memory is useless, since users arriving from each direction are leaving towards a given direction with the same probabilities.

The results show that our proposition of using memory in a mobility model siginificantly increases the accuracy of the model in cases where the ID distribution in an arbitrary cell has high variance, or has periodicities without stationary distribution.

To increase the accuracy with the use of memory in the mobility model, we introduce our Discrete Time Markov model with memory (M_n, where n denotes the number of Markov states). The states of the model show and store where the users came from. The memory can be interpreted in two meanings. Time dimension memory shows the number of timeslots in the past that the model considers. Thus a model with m time dimensions in time t can calculate the next transtion based on the user position in $(t-m, \ldots\ t-2, t-1)$. Direction dimensions memory shows the number of directions that the model can differentiate. In general a cell cluster consists of hexagonal cells. The direction dimension of a model on this cluster is maximum 6. If transition from the central cell to two adjacent cells are not differentiated then the direction dimension is decreased by 1.

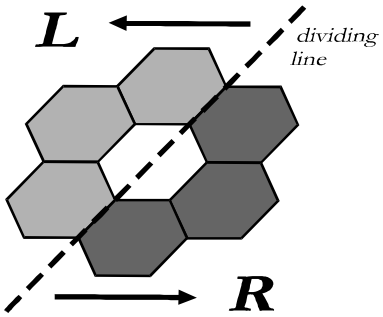


Fig. 5. Neighbour cells separated into two groups.

Obviously with increasing the dimensions of time and direction memory exponentially increases the complexity of the model, contrarily the logarithmical increase of accuracy. Our work was motivated to find an optimal size of estimation parameter space with the highest accuracy beside tolerable complexity. In the following subsections we will give example models with different time and direction dimension. We will show the coherence of the complexity and accuracy in a model with a given statespace.

4.4. Markov model extensions

In our method the direction of a user is identically distributed between 0 and 2π. The user's speed is between 0 and $V_{\max}$. After moving in a direction with a randomly chosen speed for a given Δt time, the user changes its direction and speed. N_{cell} denotes the number of cells in the network.

In a simple Markov-chain based model a user can be located in three different states during each time slot, the stay state (S) and the left-move state (L) and the right-move state (R). This classification of cells can be seen in Fig. 5.

Let us define $X(t)$ random variable, which represents the movement state of a given terminal during time slot t. We assume that $\{X(t), t = 0, 1, 2,...\}$ is a Markov chain with transition probabilities p, q, v.

We assume that a user is in cell i at the beginning of a timeslot. If the user is in state S, similarly to the previous model, it remains in the given cell in the next slot. If the user is in state R, it moves to one of the cells on the right-hand side, if in state L, it moves to the left-hand side of the dividing line (direction dimension is 3).

Since the transition propensities are not symmetric, the left-move state and the right-move state have different probabilities. Figure 6 depicts the Markov chain and transition (II) matrix.

Transition probabilities p, q and v can be determined based on the network parameters.

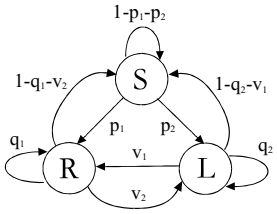
The balance equations for this Markov chain are given in Eq. (3).

$$\begin{aligned} P_S \cdot (p_1 + p_2) &= P_L \cdot (1 - q_1 - v_2) + P_R \cdot (1 - q_2 - v_1) \\ P_L \cdot (1 - q_1) &= P_S \cdot p_1 + P_R \cdot v_1 \\ P_R \cdot (1 - q_2) &= P_S \cdot p_2 + P_L \cdot v_2 \end{aligned} \tag{3}$$

We also know that $P_S + P_L + P_R = 1$ thus the steady state probabilities can be calculated.

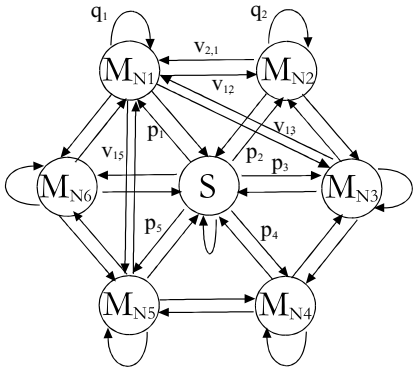
With knowledge of the result we can predict the number of mobile terminals for time slot $t + 1$ for each cell, using Eq. (4), where $S^i_{adj(r)}$ is the set of right neighbours of cell I.

$$N_i(t+1) = N_i(t) \cdot P_S(i) + +\frac{1}{3} \sum_{l, C_l \in S^i_{adj(l)}} N_l(t) \cdot P_R(l) + \frac{1}{3} \sum_{r, C_r \in S^i_{adj(r)}} N_r(t) \cdot P_M(r) \tag{4}$$



$$\Pi = \begin{bmatrix} 1-p_1-p_2 & p_1 & p_2 \\ 1-q_1-v_2 & q_1 & v_2 \\ 1-q_2-v_1 & v_1 & q_2 \end{bmatrix}$$

Fig. 6. State diagram and Ï matrix of 3-state M-model (M_3).

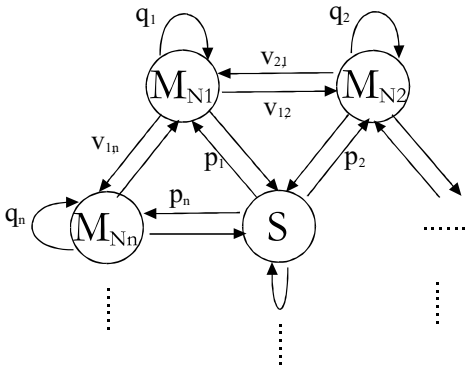


$$\Pi = \begin{bmatrix} 1-\sum_{m=1}^{6} p_m & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \\ 1-q_1-v_{1,2}-v_{1,6} & q_1 & v_{1,2} & v_{1,3} & v_{1,4} & v_{1,6} & v_{1,6} \\ 1-q_2-v_{2,3}-v_{2,1} & v_{2,1} & q_2 & v_{2,3} & v_{2,4} & v_{2,5} & v_{2,6} \\ 1-q_3-v_{3,4}-v_{3,2} & v_{3,1} & v_{3,2} & q_3 & v_{3,4} & v_{3,5} & v_{3,6} \\ 1-q_4-v_{4,5}-v_{4,3} & v_{4,1} & v_{4,2} & v_{4,3} & q_4 & v_{4,5} & v_{4,6} \\ 1-q_5-v_{5,6}-v_{5,4} & v_{5,1} & v_{5,2} & v_{5,3} & v_{5,4} & q_5 & v_{5,6} \\ 1-q_6-v_{6,1}-v_{6,5} & v_{6,1} & v_{6,2} & v_{6,3} & v_{6,4} & v_{6,5} & q_6 \end{bmatrix}$$

Fig. 7. State diagram and Π matrix of seven-state Markov model.

As we mentioned earlier this model performs well when the user's movement has only one typical direction, because in this case the handover intensities of the right-move cells does not differ significantly. It is the same in the aspect of the left-move cells.

If we try to predict the user's distribution in a city having irregular, dense road system, or in a big park where people are able to move around then the handover intensities could differ thus the calculations above could produce errors. From this point of view the best way is if we represent all of the neighbour cells as a separated Markov state, so we create 7 states, because we assume 7 elements in a cluster (direction dimension is 7) :



$$\Pi = \begin{bmatrix} 1-\sum_{m=1}^{n} p_m & p_1 & p_2 & \dots & \dots & \dots & p_n \\ 1-q_1-v_{1,2}-v_{1,n} & q_1 & v_{1,2} & \dots & \dots & \dots & v_{1,n} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 1-q_i-v_{i,i+1}-v_{i,i-1} & \dots & \dots & v_{i,i-1} & q_i & v_{i,i+1} & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 1-q_n-v_{n,1}-v_{n,n-1} & v_{n,1} & \dots & \dots & \dots & v_{n,n-1} & q_n \end{bmatrix}$$

Fig. 8. State diagram and Π matrix of n+1-state Markov model (M_{n+1}).

– stationary state (S)
– move to neighbour 1..6 state ($M_{N1}..M_{N6}$)

The Markov chain and transition matrix are more complex as it can be seen in Fig. 7. As in the previous cases the steady state probabilities can be evaluated.

To calculate the number of users in C_i for time slot t, then Eq. (5) is to be used, where the neighbour cells of cell i are indexed from 1 to 6.

$$N_i(t+1) = N_i(t) \cdot P_S(i) + \sum_{j, C_j \in S^i_{adj}} N_j(t) \cdot P_{M_{Nj+3 \bmod 6}}(l) \tag{5}$$

It is to be taken into account that in the real networks a cell does not always have six neighbours depending on the coverage. This model has to be generalized for a common case when a cell has $n$neighbouring cells. We expanded our previous mentioned model to n-neighbour case (Fig. 8), when all the n neighbours are represented with a Markov state:

– stationary state (S)
– move to neighbour 1..n state ($M_{N1}..M_{Nn}$)

The steady state probabilities can be calculated as in the previous cases Eq. (6).

$$P_S \cdot \sum_{i=1}^{n} p_i = \sum_{j=1}^{n} (1 - q_j - \sum_{l \neq k}^{n} v_{k,l}) P_{Ni}$$

	U_1	U_2	U_3	...
t_1	C_i	C_j	C_j	...
t_2	C_k	C_k	C_j	...
t_3	C_k	C_l	C_j	...
t_4	C_j	...	...	...

Fig. 9. The format of the result trace (U_x means the user x).

$$\begin{array}{l} .... \\ P_{Nk}(1-q_k) = P_S \cdot p_n + \sum_{i \neq k} P_{Ni} \cdot v_{i,k} \quad 1 \leqslant k \leqslant n \\ ... \end{array} \tag{6}$$

Using the result the predicted number of users in the next time slot is given in Eq. (7), where $P_{M_i}(j)$ denotes the steady state probabilities of moving to the cell i from cell j.

$$N_i(t+1) = N_i(t) \cdot P_S(i) + \sum_{j, C_j \in S^i_{adj}} N_j(t) \cdot P_{M_i}(j) \tag{7}$$

4.5. Determining the parameters and calculating the error of the model

In this chapter we present the transition probabilities and its determination of our Markov model. We introduce a method to process the network traces. In this section we show the error of the module based on the parameter determination.

The logs from the network provider contain the handover information and current time. Namely the handover information consist of the user ID and a source and destination cell. We split the examined interval into Δt parts. In every timeslot a cell is appended to every user. This results the input for the determination algorithm (Fig. 9).

First of all, the transition probabilities, used in previous section, are defined as:

$p_n =$ the probability of moving from stationary state (S) to M_n
$q_n =$ the probability of staying in a move state (M_n)
$v_{n,m=}$ the probability of moving from a move state (M_n) to another one (M_m)

These values are calculated based on above mentioned result trace. Let us introduce the following terminologies to process the traces and to determine the probabilities mentioned above:

Pa(X,Y)

– A two-step move pattern (Fig. 10) from C_i to C_j, in other way a user, who is in C_i at time t, and in C_j at time t+1. This trace could be used for ExtRW and HOV parameter determination also, in fact this is the "memory less trace"; we don't know, where the user came from, we only know is where it is heading to : *Pa*($C_{i,}C_j$).

Pa(X,Y,Z)

– A three-step move pattern (Fig. 10) from C_h to C_j, across C_i, in another way a user, who is in C_h at time t-1, in C_i at time at t, and in C_j at time t+1. This trace could be called a "memory trace", because during we are in C_i, we know, that the user came from C_h): *Pa*($C_{h,}C_i, C_j$).

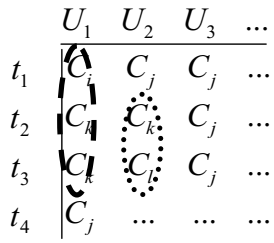


Fig. 10. Example for the move patterns. Dashed line: three-step move pattern, Dotted line: two-step move pattern.

– For all users and for all timeslots we search the *Pa*($C_{a,}C_b$) pattern in the trace . The number of the found patterns:
Num(*Pa*($C_{a,}C_b$)).
– The index of a cell, which is in z direction from the current C_j: *Ind*(*z,i*).

Using these determinations the probabilities belonging to C_j can be given by the following way from real network traces:

$$\hat{p}_{(j)\ i} = \frac{Num(Pa(C_j, C_j, C_{Ind(j,i)}))}{\sum_{l=0}^{n}\sum_{k=0}^{n} Num(Pa(C_{ind(j,l)}, C_j, C_{ind(j,k)})) = Sum[j]} \tag{8}$$

$$\hat{q}_{(j)\ i} = \frac{Num(Pa(C_{ind(j,i)}, C_j, C_{Ind(j,i)}))}{Sum[j]} \tag{9}$$

$$\hat{v}_{(j)\ i} = \frac{Num(Pa(C_{ind(j,i)}, C_j, C_{Ind(j,i)}))}{Sum[j]}, \tag{10}$$

Based on these equations a prediction can be given for the users distribution. The question is the accuracy of the predicting model.

It can be seen that these $\hat{p}_i, \hat{q}_i, \hat{v}_{i,k}$ values were defined based on relative frequency in the network traces. Accuracy of this calculation depends mainly on the number of samples. That is a finite set of samples could not be sufficient, there is always a minimal error which is the difference between the real, model ($p_i, q_i, v_{i,k}$) and the calculated ($\hat{p}_i, \hat{q}_i, \hat{v}_{i,k}$) values. Let us define the ε_p, εq,ε_v as the error of $p_i, q_i, v_{i,k}$ determination of the parameters of the Markov model (As ε_p in Eq. (11), $\varepsilon_{q,}$ ε_v can be determined similarly).

The mean value of calculated parameters (for example $E(\hat{p}_i)$) is the real value because of the law of averages.

$$\varepsilon_p = |\hat{p}_i - p_i| = |\hat{p}_i - E(\hat{p}_i)| \rightarrow p_i = \hat{p}_i + \varepsilon_p \tag{11}$$

As we defined earlier the n-state Markov model (see Chapter 2.3.1) satisfies the general matrix equation $P = P\Pi$, which can be solved. But we are not able to determine exactly the Π matrix. Calculating with the relative frequencies we get a $\hat{\Pi}$ matrix, which estimates the Π matrix with calculation errors. When we solve the matrix equation $P = P\Pi$ using $\hat{\Pi}$ instead of Π, we get a $\hat{P}$ equals P with the addition of the model error. Summarizing these coherences:

– theoretical solution: $P = P\Pi$

– practical solution: $\hat{P} = \hat{P}\hat{\Pi}$

$$\hat{P} = P + H \tag{12}$$

$$\hat{P} = \hat{P}\hat{\Pi} \rightarrow P + H = (P + H)(\Pi + \mathrm{E}) \tag{13}$$

where H denotes the error of the steady state probability comeing from the parameter calculation error, and E means the matrix derived from the ε_p, ε_q, ε_v values. Our aim by the following is to calculate the dependency of the model error on the network parameters.

Let us start from the Eq. (6). described with the calculated parameters instead of the real parameters Eq. (14).

$$\begin{aligned}
&\hat{P}_S \cdot \sum_{i=1}^{n} \hat{p}_i = \sum_{j=1}^{n} \left(1 - \hat{q}_i - \sum_{l \neq j}^{n} \hat{v}_{j,l}\right) \hat{P}_{Ni} \\
&\quad .... \\
&\hat{P}_{Nk}(1 - \hat{q}_k) = \hat{P}_S \cdot \hat{p}_n + \sum_{i \neq k} \hat{P}_{Ni} \cdot \hat{v}_{i,k} \;\; 1 \leqslant k \leqslant n \\
&\quad ...
\end{aligned} \tag{14}$$

Replace the variables by Eq. (12) and 11 to get the following equation Eq. (15).

$$\begin{aligned}
&(P_S + H_S) \cdot \sum_{i=1}^{n} (p_i + \varepsilon) = \sum_{j=1}^{n} (1 - (q_i + \varepsilon) - \sum_{l \neq j}^{n} (v_{j,l} + \varepsilon))(P_{Ni} + H_{Ni}) \\
&... \\
&(P_{Nk} + H_{Nk})(1 - (q_k + \varepsilon)) = (P_S + H_S) \cdot (p_n + \varepsilon) + \sum_{i \neq k} (P_{Ni} + H_{Ni}) \cdot (v_{i,k} + \varepsilon) \\
&... \\
&1 \leqslant k \leqslant n
\end{aligned} \tag{15}$$

As next step we denote H_S, H_{Nk} using the terms from Eq. (6). and the coherence $\varepsilon = \max(\varepsilon_p, \varepsilon_q, \varepsilon_v)$:

$$\begin{aligned}
&H_S = \sum_{j=1}^{n} (1 - q_i - \sum_{l \neq j}^{n} v_{j,l}) H_{Ni} + (1 - \sum_{i=1}^{n} p_i) H_S - n\varepsilon(nH_{Ni} + 1 + H_S) \\
&... \\
&H_{Nk} = H_S p_m + \sum_{i \neq k} H_{Ni} \cdot v_{i,k} + H_{Nk} q_k + H_S \varepsilon + (nH_{Nk} + 1)\varepsilon \\
&... \\
&1 \leqslant k \leqslant n
\end{aligned} \tag{16}$$

We have an equation system with $n + 1$ equation Eq. (16). Let us do some simplifying by calculating average of some parameters in the following way:

$$\begin{aligned}
&p = avg(p_i) \;\; \forall i \\
&q = avg(q_i) \;\; \forall i \\
&v = avg(v_{i,j}) \;\; \forall i, \forall j
\end{aligned} \tag{17}$$

These three general probability parameters can describe typical user movements in the current cell.

Parameter p means the probability of the moving after stop, q is for describing how the user can hold its moving direction, and at last v, the opposite of q, namely how often the moving direction of the users changed.

Using these general parameters yields an upper estimation with an equation system in 2 variables:

$$\begin{aligned} H_S &= n(1-q-(n-1)v)H_N + (1-np)H_S - n\varepsilon(nH_N + 1 + H_S) \\ H_N &= H_S p_m + (n-1)H_N v + H_N q_+ H_S \varepsilon + (nH_N + 1)\varepsilon \end{aligned} \tag{18}$$

If we examine the ε error rate, we find that an upper estimation can be applied for it as well using the Chebyshev inequality. The variance of relative frequency is well known:

$$\sigma(\hat{p}) = \sqrt{\frac{p(1-p)}{m}} \tag{19}$$

where $p = E(\hat{p}_i)$, and m are the number of samples in the trace.

Let us make an upper estimation on Eq. (19) to eliminate variable p:

$$\sqrt{\frac{p(1-p)}{m}} \leqslant \sqrt{\frac{1}{4m}} = \frac{1}{2\sqrt{m}} \tag{20}$$

Using this result in the Chebyshev inequality:

$$P(|\hat{p} - E(\hat{p})| \geqslant \varepsilon) \leqslant \frac{1}{2\sqrt{m}\varepsilon^2} \tag{21}$$

If we assume that $|\hat{p} - E(\hat{p})|$is not greater than ε with 99% probability, and that $m = 10000$, we reach an upper estimation for ε:

$$\begin{aligned} &0.01 \leqslant \tfrac{1}{2\sqrt{m}\varepsilon^2} \\ &\ldots \\ &-\tfrac{1}{\sqrt{2}} \leqslant \varepsilon \leqslant \tfrac{1}{\sqrt{2}} \rightarrow \varepsilon = \tfrac{1}{\sqrt{2}} \end{aligned} \tag{22}$$

In order to get an equation system that depends only on p, q, v, the ε is replaced in Eq. (18).

$$\begin{aligned} H_S &= n(1-q-(n-1)v)H_N + (1-np)H_S - n\frac{1}{\sqrt{2}}(nH_N + 1 + H_S) \\ H_N &= H_S p_m + (n-1)H_N v + H_N q + H_S \frac{1}{\sqrt{2}} + (nH_N + 1)\frac{1}{\sqrt{2}} \end{aligned} \tag{23}$$

H_S, H_N, the error of the steady state probability can be calculated from the equation system, but due to the limitation of this paper we do not present it in detail.

As the last step we define H_{Model}, the model error from H_S, H_N: $H_{Model} = H_S + nH_N$

Using the plotting abilities of *Mathematica* [16] we examined this model error depending on p, q, v, n. The next figures depict the results. The x axis shows the number of states (n), the y axis represents the error of the module.

We analyzed four different scenarios considered as typical user movements. The result of the first combination is depicted on Fig. 11. In this case the all of the general transition parameters (p, q, v) take low values, meaning that the users in the current cell move slowly, stop several times. The function of model error increases linearly, as it is depicted.

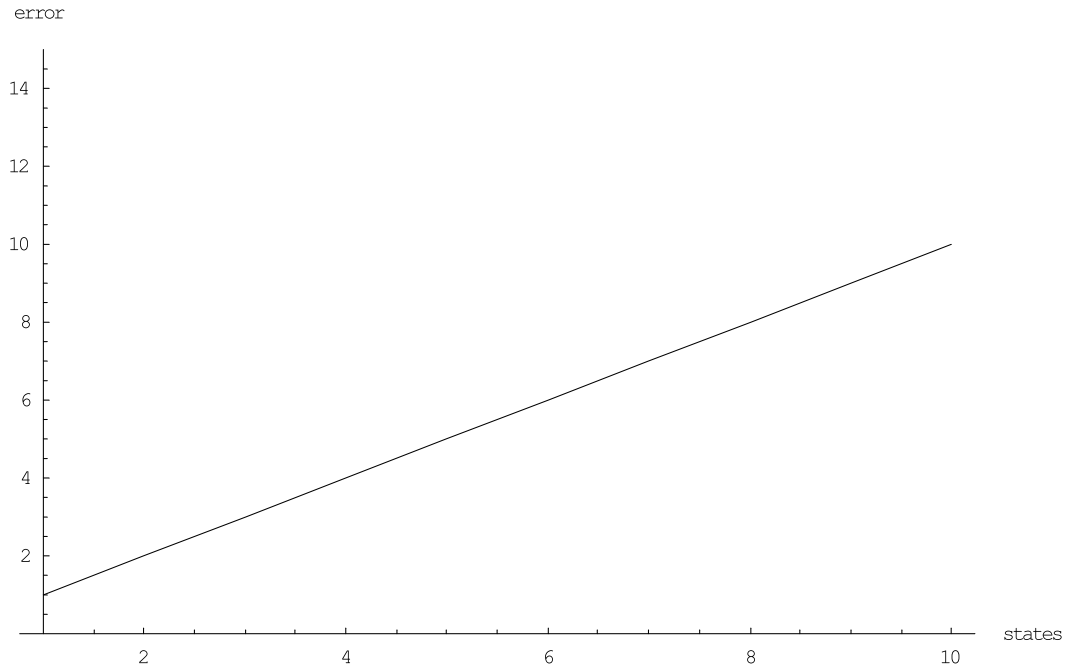


Fig. 11. Model error, $p = 0.1$, $q = 0.1$, $v = 0.1$; Slow motion in current cell.

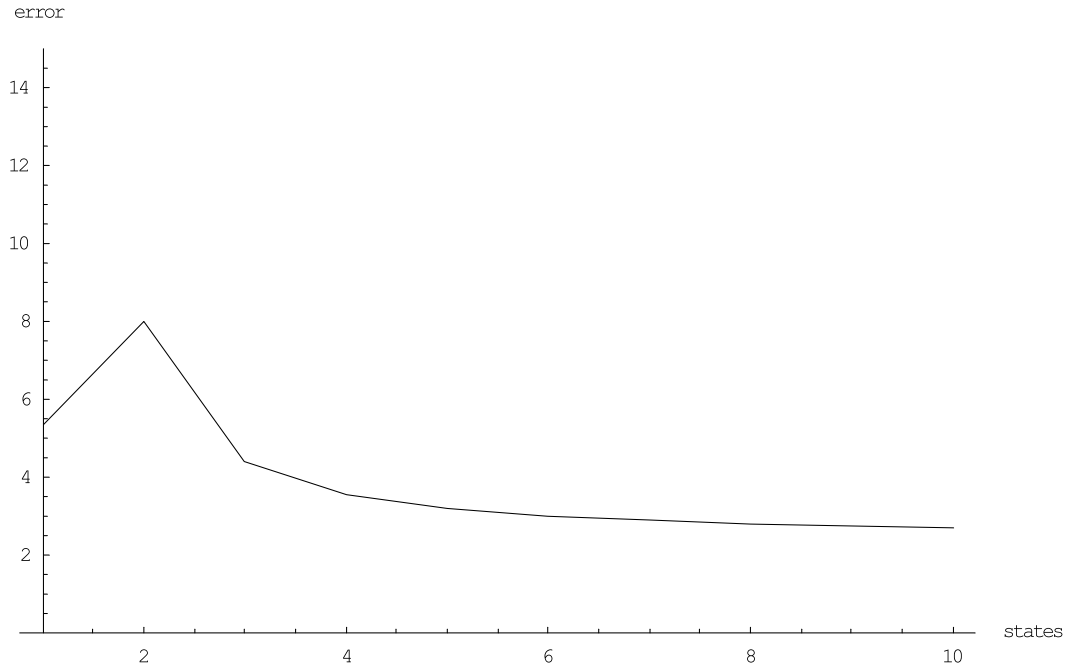


Fig. 12. Model error, $p = 0.2$, $q = 0.9$, $v = 0.9$; Fast motion with direction changes and direction holding as well.

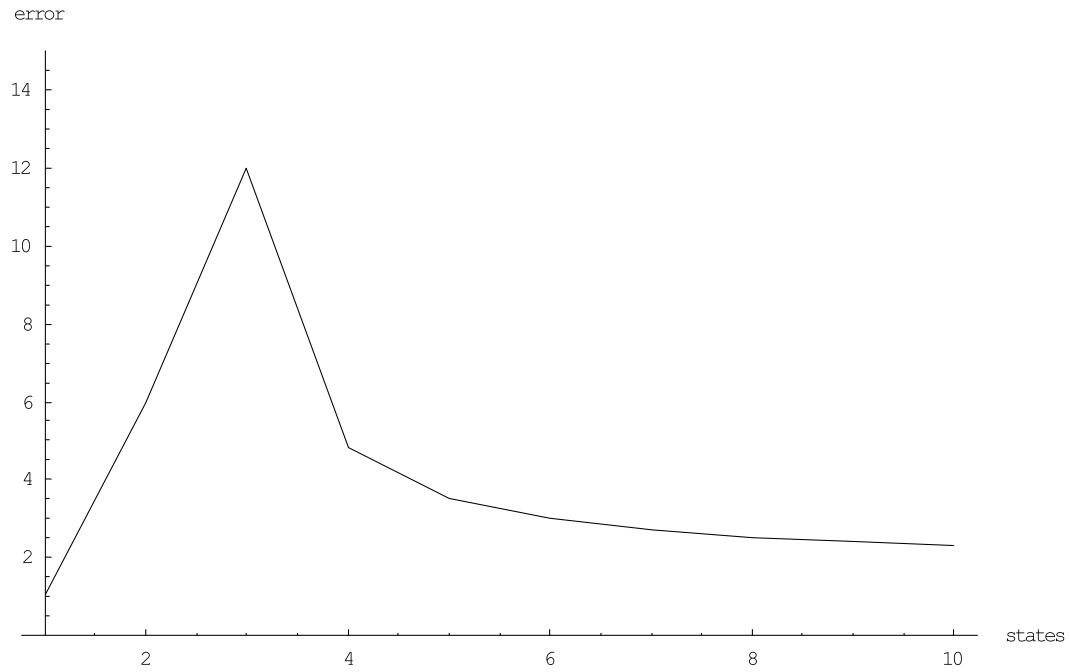


Fig. 13. Model error, $p = 0.2$, $q = 0.1$, $v = 0.9$; Fast motion with direction changes.

The second scenario shows the opposite of the previous case, namely fast movement in all direction, none of the users stops (Fig. 12).

Third case simulates a fast, direction-changing movement environment. The function on Fig. 13 has a local minimum in $x = 3$, after that it is decreasing logarithmical.

In the last situation the users move fast, and hold their direction. The model error increasing slowly (Fig. 14).

In this section we determined the previously introduced Markov model parameters from the network traces, defined that the model error derived from the parameter calculations, and examined it dependently on the user movement behaviors.

4.6. Complexity and accuracy of the extended markov model

In the previous section we introduced a Markov model generator method, an optional model can be derived depending on the resource requirements (complexity) and the demanded precision (accuracy).

The accuracy of the model is increasing in function of the number of states. The number of states is increasing when the memory (time dimension) is increased, or when the number of direction (direction dimension) is increased. Time dimension increases the states exponentially, direction dimension increases it linearly. With the state-space increasing, the computational complexity of the Markov steady state calculations are also following a rising curve. The question is the characteristic of these functions and the existence of a theoretical or practical optimum point.

It is assumed that each cell has N neighbors and the 3-state (stay, left and right-move state) model is used to determine the user movement. It is also assumed that N/2 cells belong to both left and right Markov-states, and the users are uniformly distributed between cells. A theoretical error can be derived

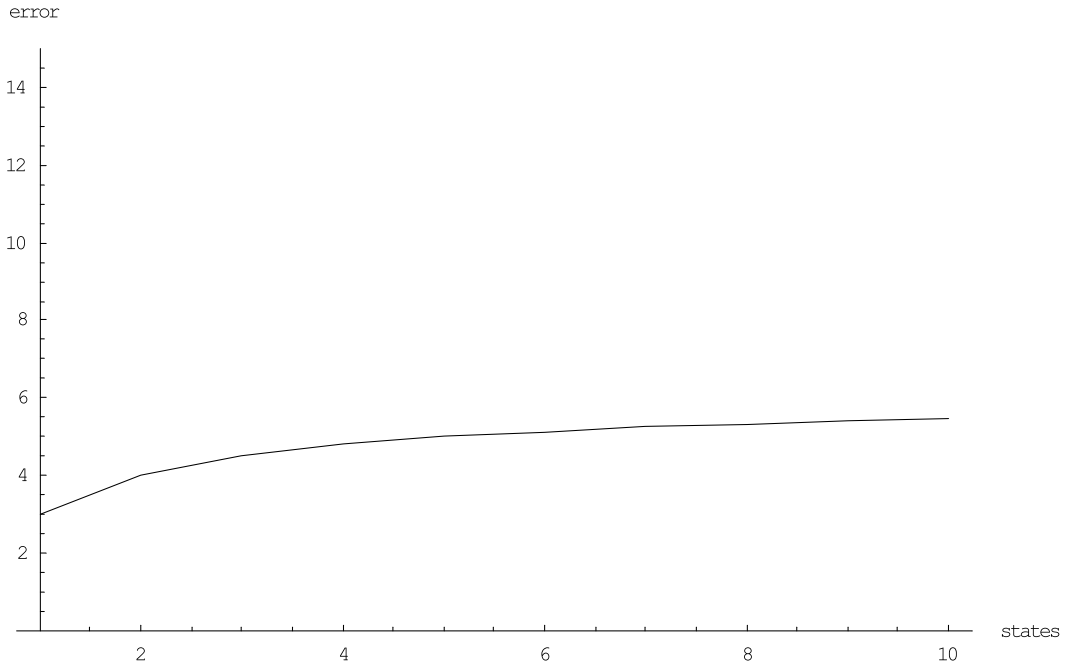


Fig. 14. Model error, $p = 0.3$, $q = 0.9$, $v = 0.1$ Fast motion with direction holding.

from this assumption since in most cases the user motion pattern does not result in a uniform distribution in the N/2 cells. In the worst case the users move with 1 probability into one of the neighbouring cells. In our estimation calculations the error can be measured with the difference between the uniform distribution and the worst case. This difference in mathematical equation is the following:

$$1 - \frac{1}{N/M} + \frac{1}{N/M} * ((N/M) - 1), \tag{24}$$

where N means the neighbour numbers, and M means the direction numbers in the model.

We measured the computation complexity also as the function of state number. This enables us to compare complexity and prediction error in an easy way. Based on the 1..M-state model the prediction computation is calculated with the costs of Markov steady state mathematical operations and other procedures necessary for transition probabilities. The complexity can be estimated with o($M^3_{\text{states}} + M_{\text{states}}$ + 1/ M_{states}).

Figure 15 shows the complexity and error characteristic. In the given model calculations the optimal point of operation is around 5 states where error is minimal at this level of complexity.

In the previous comparison the direction dimension sizing is used only. If we want to look back for the estimation, than memory has to be introduced into the model. In fact this means that every state in the Markov chain has to be changed with M states. This causes exponential state number explosion that can be seen on the Fig. 16.

As we have extended our model step by step in time and direction dimension, its precision increased as well as the complexity of the model. In order to decrease it, we developed a simple algorithm for direction dimension, which is able to minimize the number of states based on merging adjacent cells. The input parameters are the following:

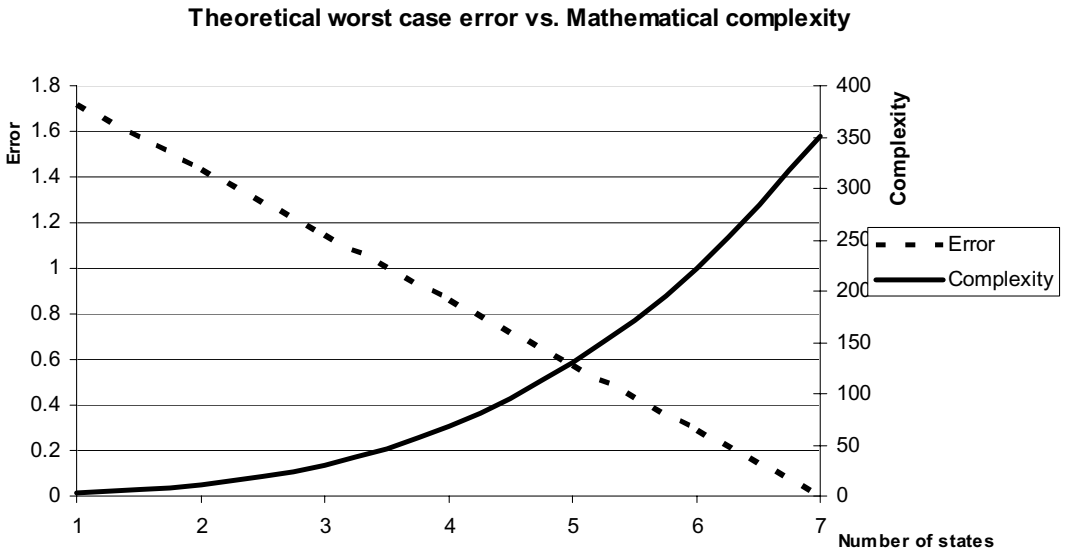


Fig. 15. Complexity (solid) and accuracy of the calculation (dashed).

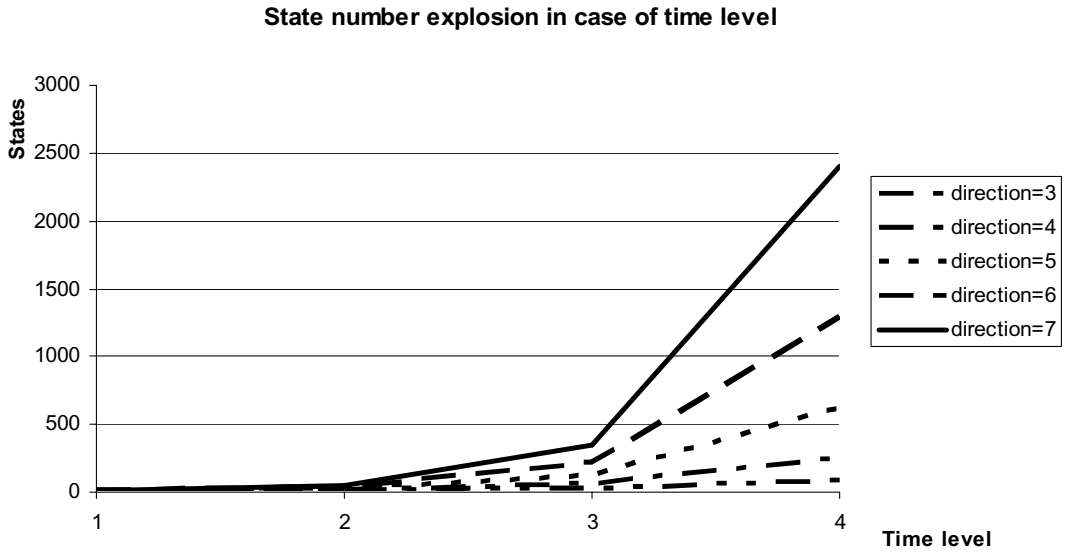


Fig. 16. The increase of number of states in case of different direction dimensions (3,4,5,6,7 states), and different time dimensions (1,2,3,4).

– an acceptable *Err* rate of error caused by the merging
– h_{kl} elements of the handover matrix (where h_{kl} is the handover probability from C_l to $C_{k,}\forall C_k, \forall C_l \in S^i_{adj}, k, l = 1 \ldots N_{cell}$). $Q_{Hi}(n_i+1 \text{ x } n_i+1)$ matrix is defined (where n_i is the number of neighbour cells of cell i). The elements in the first row and column are the handover probabilities of the examined cell i.

Based on the Q_{Hi} matrix (elements denoted by h^i_{kl}) and the error rate a group of cells is to be found, where the absolute difference between h^i_{ki} and h^i_{li}, and between h^i_{ik} and h^i_{il} ($\forall C_l$, $\forall C_k$ in a group) is less

than *Err*.

The algorithm is given with its pseudo code:

```
original_Q = QHi;
minimal_grouping_Q = QHi;
minimal_grouping_num = ni;
For x = 2 to ni +1 {
  grouping.clear();
  nullstations.clear();
  QHi = original_Q;
  For j = x − 1 to ni + x − 1 {
    k = (j mod 7) + 1;
    If ((k! = x) && (k! = i) && (!grouping.member(k))) {
    group_row = hik_
    group_column = trans(hi_k);
    temp_g_row = 0;
    temp_g_column = trans(0);
    group[k].create();
    group[k].add(k);
    For l = 2 to ni {
      if (!grouping.member(l)) {
      greater_than_Err = false;
      For m = 1 to ni{
        if((!nullstation.member(m)) && (m! = l)){
        if((|hikm- hilm | >Err) || (|himk- himl | >Err)){
        greater_than_Err = true;
        break;
        } else{
        temp_g_row[m]=(|group_row[m]+ hilm |/2);
        temp_g_column[m]=(|group_column[m]+ himl |/2);
  }}} if (!greater_than_Err){

    nullstations.add(m);
    group[k].add(l);
    group_row = temp_g_row;
    group_column = temp_g_column;
  }}}
  QHi[k,_] = group_row;
  QHi[_,k] = trans(group_column);
  }}
  if(grouping.length<minimal_grouping_num) {
  minimal_grouping_Q = QHi;
  minimal_grouping_num = grouping.length;
}}
```

As a result of the algorithm the simplified equation is Eq. (25), where K denotes a group of cells, m_K^j the number of elements in group K, and $P_{M_K}(j)$ is the steady state probability of moving to group K

from cell j.

$$N_i(t+1) = N_i(t) \cdot P_S(i) + \sum_{j, C_j \in S^i_{adj}} N_j(t) \cdot \frac{1}{m^j_K} P_{M_K}(j) \qquad C_i \in K \tag{25}$$

In this section we examined complexity and theoretical errors. We exhibited the state number explosion at time dimension increasing. Chapter 3. contains measurements with simulations, which confirms our mathematical results.

5. Path based mobility model

At the definition of the prediction the importance of the time dimension, besides the direction dimension, was mentioned at the introduction and detailed presentation of the Markov model. At the Markov model we treated the m depth time dimension of the users arriving from different directions uniformly, time dimension being the number of steps that was taken into account from the user's past. As we could see with the inclusion of each new step, the number of states increased exponentially. However treating this factor generally might not be optimal. Let us check if it is useful to look back in a certain direction, if yes how much, examine how much the prediction is influenced by taking a certain direction's different depths into account during the calculations. Also let us analyze how the past directions can be combined, similarly to the previously introduced state combination algorithm. In this chapter the previously introduced mobility modeling will be put into a slightly different aspect, a second model, the *path based mobility* model, including the analysis of the time dimension's each past direction and multi-level predictions will be introduced besides the Markov solution.

5.1. The basic idea of the path based mobility model

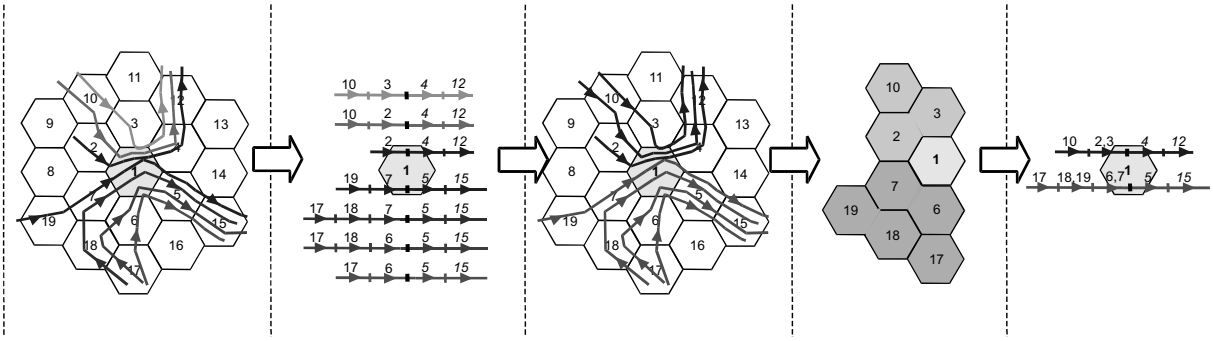
The importance of the time dimension was introduced on Fig. 3 and in the corresponding mathematical calculations. The future movement of the users is highly influenced by the path they have taken in the past until the investigated point. Leaving this out of consideration would introduce large errors into the mobility model. However it is not always useful to look back into each direction, or to look back in equal depth into each direction. One of the path based mobility model's – which gives a prediction for a given cell similarly to the Markov model – main idea is to analyze the importance of each direction traveled by the users, and decide its importance for consideration. The importance is decided based on two basic criteria:

- the rate of the users arriving from a certain direction into the cell.
- the fluctuation of the number of users arriving from a certain direction based on previous data.

Taking these and the previous steps of the users into consideration we get the different time level paths pointing into the actual cell (Fig. 17a).

In the second step we analyze which outgoing predictions correspond to the different length, incoming paths. The outgoing predictions can be examined for more future steps, contrary to the Markov model, also depending on the rate of the users leaving in a certain direction. If outgoing predictions of two incoming path matches with a certain limit, the two incoming path should not be treated separately but combined (Fig. 17c). Therefore the resulting paths consist of more slices; each slice might contain multiple cells (Fig. 17e).

As result we get reduced incoming paths and corresponding, also multi-level, different length predictions, which lead to a more accurate and less complex model.



a. Different user movements | b. Different paths with the same predictions | c. Merged user movements | d. Merged cells | e. Merged paths, the output of the model

Fig. 17. Path based mobility model.

5.2. Determine the different paths

In this chapter we present the mathematical repository and the parameters of the path based mobility model and the algorithm determining the optimal and minimal numbers of paths.

As earlier mentioned two criteria have to be examined to decide if it is useful to take, if yes in what depth, a certain direction into consideration. The first is the main value of the users in the certain cell:

$$E^{i,j}_{user\_rate} = E\left(\frac{N_{i,j}}{N_i}\right) \tag{26}$$

where i is the identifier of the cell, j is the analyzed direction, $N_{i,j}$ is the number of users from direction j into cell i, and N_i is the number of users in cell i. The second value is the variance of the number of users:

$$V^{i,j}_{user\_rate} = E\left(\left|E\left(\frac{N_{i,j}}{N_i}\right) - \left(\frac{N_{i,j}}{N_i}\right)\right|\right) = \sigma\left(\frac{N_{i,j}}{N_i}\right). \tag{27}$$

After this we got incoming paths for the actual cell. We use the simple handover model (HOV) to determine the predictions for the paths, which result in a vector p. Determination of p_i element of vector p is done by analog method with pattern matching described in Section 2.3.2, but in this case we calculate forward move patterns (Fig. 18) to determine the relative probabilities.

In the next step, not to analyze paths corresponding to different arriving directions if their predictions match, let's introduce the mean difference between predictions for two paths:

$$D^{a,b}_{pred} = \frac{\sum\limits_{k \forall n_{dir}} \left|p^a_k - p^b_k\right|}{n_{dir}} \tag{28}$$

where a, b are the two incoming paths, p^a_k is the element of HOV vector with direction k, for path a, and n_{dir} is all the possible direction. If $D^{a,b}_{pred}$ is smaller than a given parameter, that is adequately small, then the paths a and b should be combined and treated together. Let us introduce the model's parameters before the detailed description of the algorithm.

– $\varepsilon_{userr\ variance}$ – limit of variance, if $V^{i,j}_{user\_rate}$ is higher than this, than direction j corresponding to cell i has to be further analyzed.

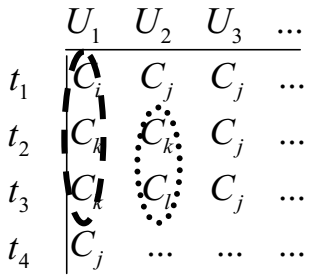


Fig. 18. Example for the move patterns for HOV determination (C is theactual cell). Dashed line: three-step forward move pattern, Dotted line: two-step forward move pattern.

- ε_{pred} – if for any k direction corresponding to path a, p_k is less than this, then the predicted path can be disregarded; taken into account when determining vector p.
- $\varepsilon_{diff\ pred}$ – limit of the mean difference between predictions for two paths; if $D^{a,b}_{pred}$ is smaller than this, paths a and b can be combined and further treated together.
- n_{past} – the maximum number of past steps
- n_{pred} – the maximum number of future steps, taken into account when determining vector p.

Beside the introduced parameters other variable types and functions belong to the algorithm:

- *group* – list of cells, container for a slice of the path.
- *path* – list of groups, container for paths.
- *get_neighbours*(*ListofCells*) – the common neigh-bours of the list of cell.
- *path.get_sector*(*depth*) – returns the cells from the depth th slice of a path.
- *get_variance_recursive*(*cell, depth*) – it analyzes the $V^{i,j}_{user\_rate}$ deviation of the users heading to cell, starting from a given cell, recursively within maximum depth steps.

Based on these the algorithm run for Cell C_a is the following:

```
group temp_group;
group temp_group_set[];
root path;
path path[];
path closed_path[];
root.add(C_a, 0);
path.add(root);
for i = 0 to n_past{
  foreach j in path_cont.get_paths(){
    neighbours = get_neighbours(j.get_sector(i));
    foreach k in neighbours {
      if !temp_group_set.is_in(k) {
      temp_group.clear();
      temp_group.add(k);
      foreach l in neighbours {
        if (k! = l) && !temp_group_set.is_in(l) {
          fitable_to_group = true;
          foreach m in temp_group {
```

if ($D^{a,b}_{pred} > \varepsilon_{diffpred}$) {
fitable_to_group = *false;*
break;}}
if (*fitable_to_group*) *temp_group.add*(*l*);}}
temp_group_set.add(*temp_group*);}}
temp_group.clear();
foreach k *in temp_group_set* {
foreach l *in k.elements*() {
if ($V^{a,neighbours\ ,j}_{user\_rate} < \varepsilon_{userrvariance}$) &&
(*get_variance_recursive*(*j*, $\left\lceil E^{i,j}_{user\_rate} \cdot (n_{past} - i) \right\rceil$)< $\varepsilon_{userrvariance}$)
k.remove(*l*);
temp_group_set.update(*k*);
temp_group.add(*l*);
}}}
temp_path.add(*temp_group*,*i*);
closed_path.add(*temp_path*);
foreach k in temp_group_set { *path_cont.add*(*k*,*i*);}
}}
foreach i *in path* {*closed_path.add*(*i*);}

In the algorithm $V^{i,S,j}_{user\_rate}$ is a reduced version of Eq. (25), when we only take the users arriving from S direction entering into cell i into account at determination of the deviation.

The algorithm with the given parameter results in minimal path numbers.

Proof: the algorithm basically performs a reduction from all possible paths, leaving out any of the steps results in more paths or the given boundary conditions are not fulfilled.

5.3. Complexity of the model

The complexity of the model is determined by the number of executed steps. Let us examine how the number of steps changes as the function of the parameters. Consider the processing of the trace file, accessing the read data and the simpler mathematical operating as one unit. The complexity of the algorithm can be determined as follows:

$$2n_{user} + n_{past} \cdot n_{nb} \cdot (n^{path}_{cell} \cdot n^{path}_{cell}(n^{path}_{pred} + 2) + n_{nb} \cdot n^{path}_{cell} + n_{nb}) \tag{29}$$

where n_{user} is the number of users in the used network trace, n_{nb} is the average number of neighbor cells from the point of the cells in one slice of the path, n^{path}_{cell} is the average number of cells in one slice of the path, n^{path}_{pred} is the average number of predicted paths for one path. The parameters defined depend on the size of the trace, the time of sampling and the mobility speed of the users. Perform the simplification, take the largest from the parameters and denote it with n.

$$n = \max(n_{past}, n_{user} n^{path}_{pred}, n^{path}_{cell}, n_{nb}) \tag{30}$$

In this case the complexity of the algorithm can be expressed the following simple way (Fig. 19):

$$o(n^5 + n^4 + n^3) \tag{31}$$

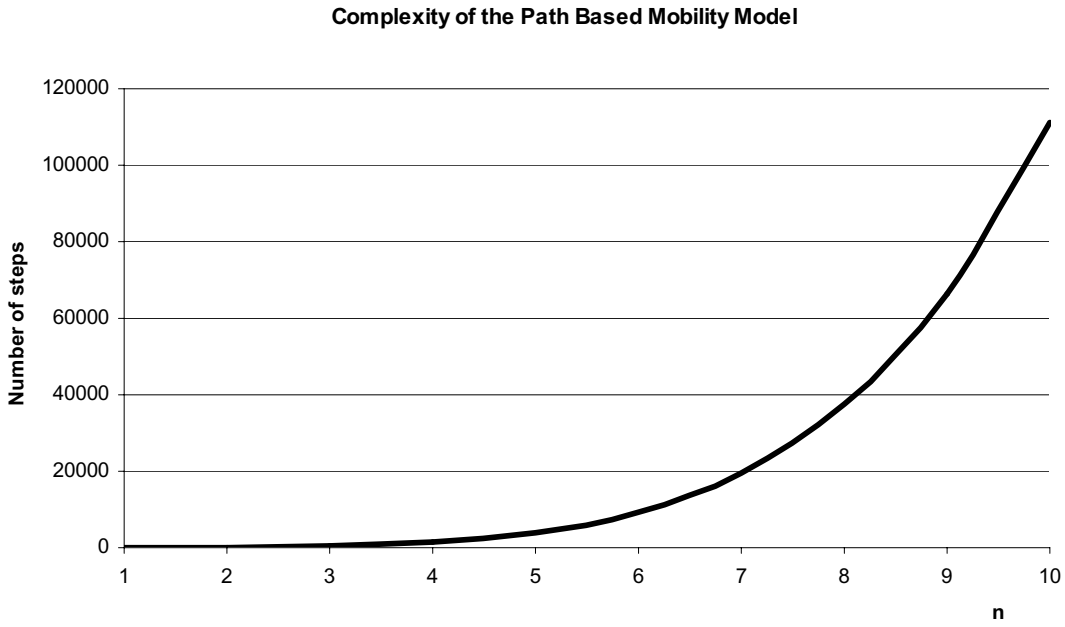


Fig. 19. Complexity of the algorithm.

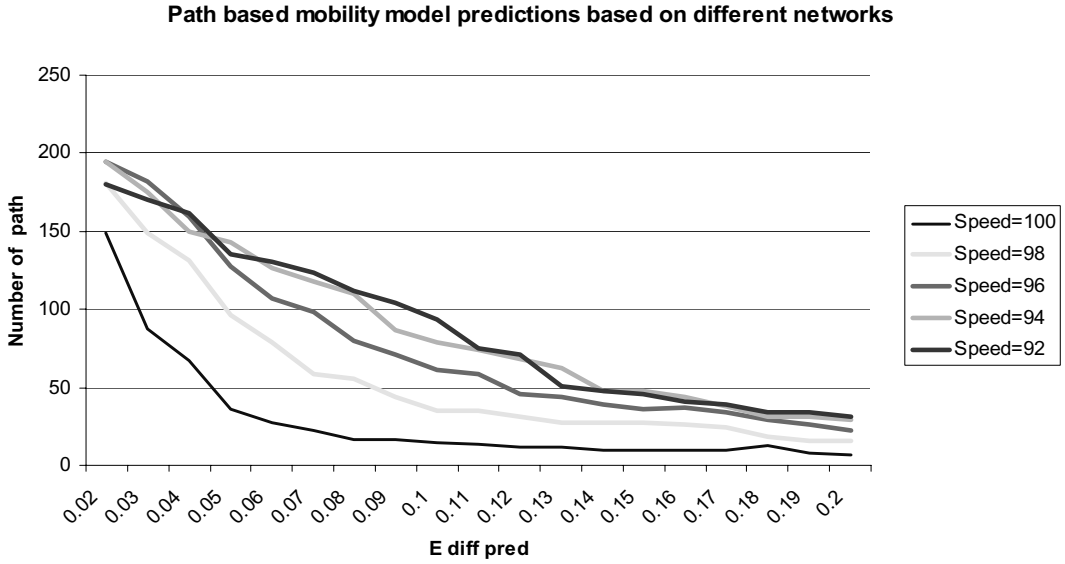


Fig. 20. Effect of the $\varepsilon_{diffpred}$ parameter with high speed.

5.4. Measures

The Path Base Mobility solution's one basic output that can be analyzed besides the actually resulting paths is the number of paths. The more paths determined by the algorithm, the more difficult the realization of the prediction will be. Taking this into consideration, it is really important that the input parameters get appropriate values as function of a defined goal.

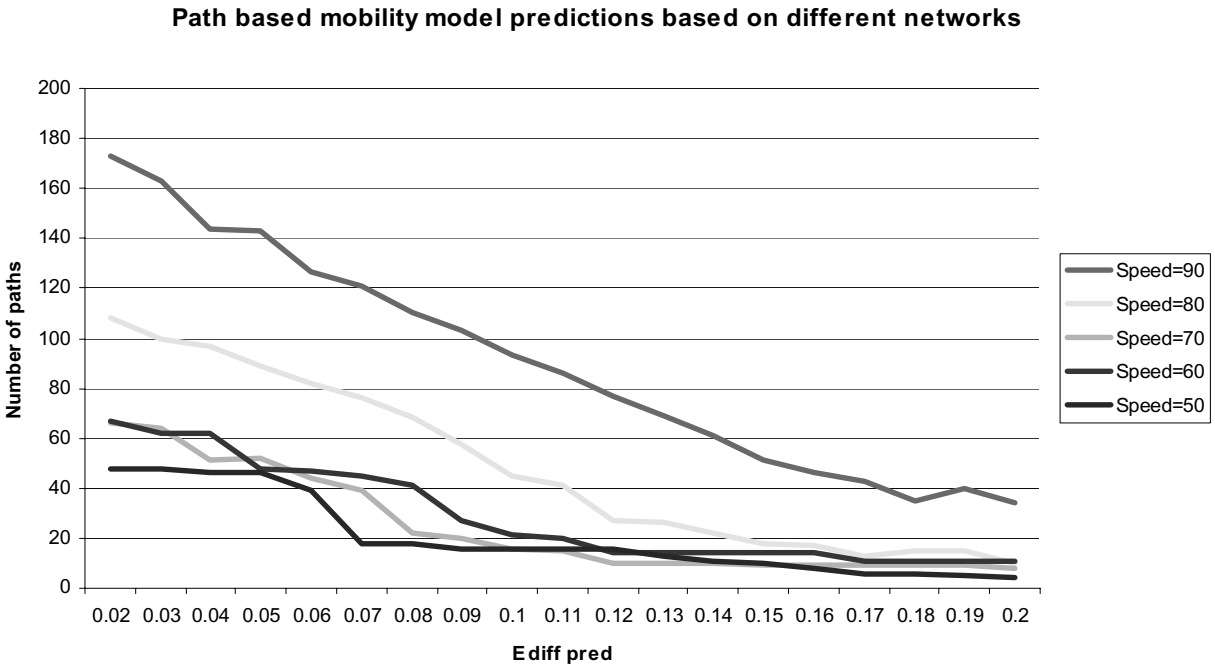


Fig. 21. Effect of the $\varepsilon_{diffpred}$ parameter with lower speed.

An erlang simulation was prepared to justify the algorithm and to monitor the effects of the parameters. The simulation was prepared based on mobility traces taken for different characteristic user speeds. The mobility traces were provided by OMNet simulation described in Chapter 6. The value of the speed was a relative number between 0 and 100, 100 meaning fast users, moving in each timeslot, 0 meaning slow, nearly steady users.

The change in the two parameters $\varepsilon_{diffpred}$ and n_{past} is shown in relation to the number of paths. Changing the other parameters does not result in significant change in the number of calculated paths.

Two separate measurements were executed to analyze the change of $\varepsilon_{diffpred}$. In the first one (Fig. 20) the algorithm was run on trace data resulting from user movements with rather higher, 100 and 92 speeds. It can be seen that curve corresponding to speed 100 strongly breaks down by increasing the parameter and always results in far less calculated paths as compared to lower speeds. By increasing the speed, the resulting curve approaches a linear curve with larger path numbers on average.

In the second measurement for $\varepsilon_{diffpred}$ (Fig. 21) the change in the number of resulting paths was examined with user movement speeds 90 and 50. The curves' characteristics are linear, but changed in relation to the number of paths. The algorithm calculated more paths for higher speeds than for lower speeds.

The curve representing the change in the number of paths in function of the speed has a local extreme at speed 90; this clearly shows on Fig. 22.

The number of paths is increased rapidly, by increasing the limit of looking back into the past. This is shown on Fig. 23.

The path based mobility model, the corresponding algorithms were introduced, its complexity and a few simulation results were discussed in this chapter.

In the next chapter the accuracy of the created models is proved by simulation results in contrary to the other models in the bibliography. The Markov model can be considered as a certain specialization

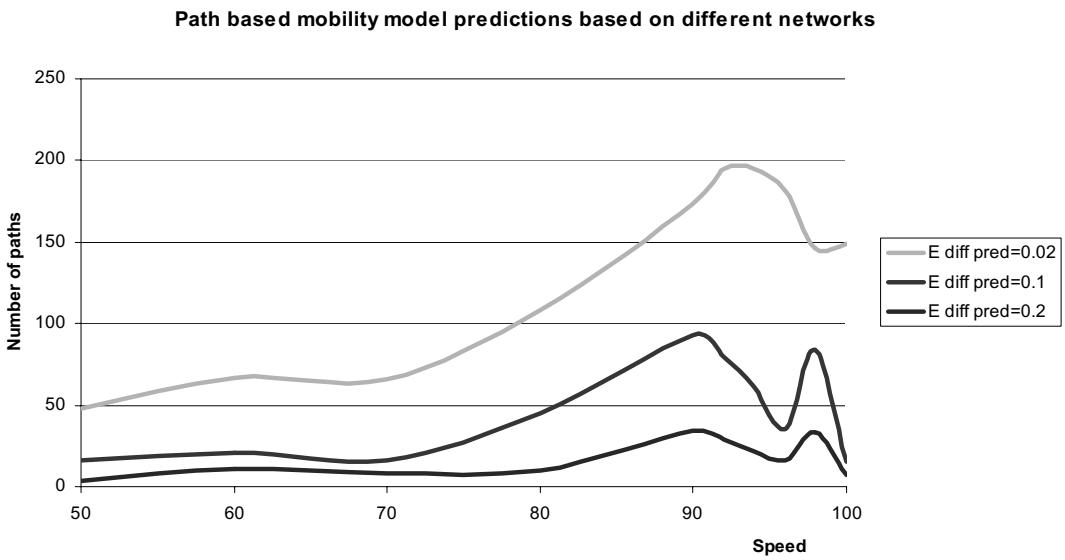


Fig. 22. Relationship between speed and number of path with different $\varepsilon_{diffpred}$ parameters.

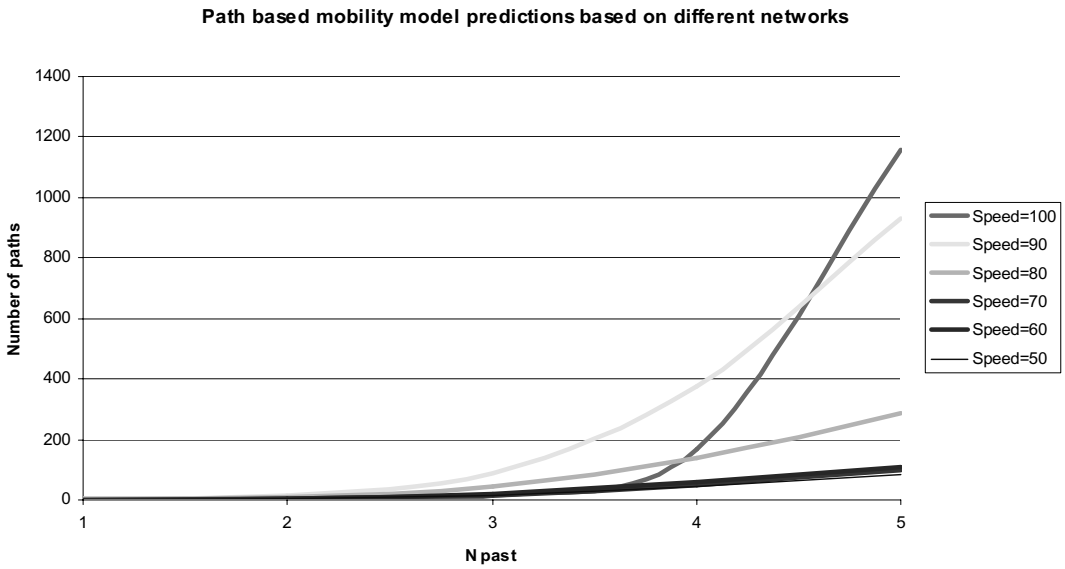


Fig. 23. Effect of the n_{past} parameter.

of the path based mobility model. Due to this, the results of the Markov model in the simulation chapter also prove the path based mobility model as well.

6. Accuracy measurements by simulation

In this chapter we compare the accuracy of the proposed extensions and the Path Based mobility solution to other models can be found in the literature. The Random Walk-based mobility models

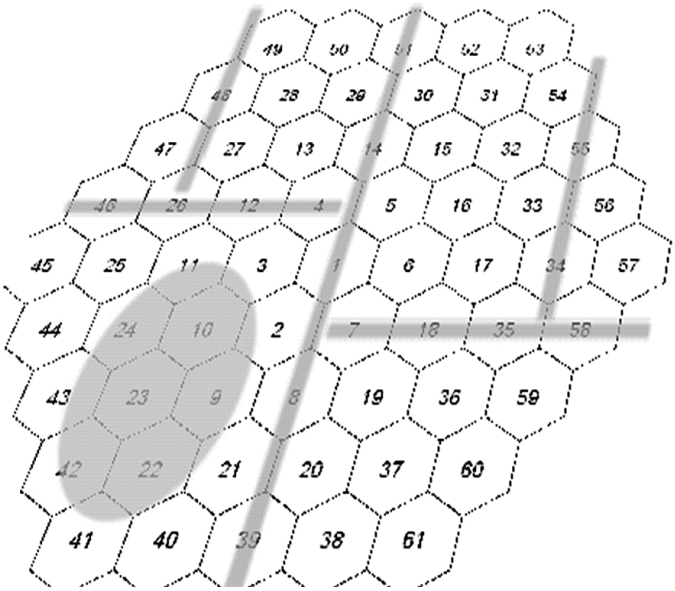


Fig. 24. Cell cluster with streets and park.

depend on the transition probabilities. The RW model is only capable of accurate prediction of user movements in case of uniform movement distributions (e.g. all elements in the transition probability vector are equal to 1/6).

According to the calculations and simulations, the Markovian approach of user movement produces better estimation of the users' distribution in a cell cluster.

The estimation procedure was validated by a simulation environment of a cell cluster shown on Fig. 24.

The simulation was written in the open source OMNet++ (Fig. 25) using C++ language. The cluster consisted of 61 named cells, the simulation environment included geographical data that is interpreted as streets on the cluster area. The drift of the movement is heading to the streets from neutral areas.

The simulation used 610 mobile terminal (10 for each cell), in the initial state uniformly distributed in the cluster. The average motion velocity of the users is parameterized with a simple PH cell dwell time simulator (reciprocal of exponentially distributed values).

The simulation consists of two parts. The trace simulation is the series of cell-transitions that the mobiles have initiated. It produces a time-trace that contains the actual location data for each mobile terminal in the network. We have used this trace simulation as if it was a provider's real network trace.

We described estimation procedures based on the different mobility model discussed in this paper. The first reference model is the simple Random Walk (RW) with constant speed and uniformly distributed direction-probabilities. The RW estimation draws a direction out of the possible six in every timeslot for every simulated user, and forwards the given user to the adjacent cell in the drawn direction.

A widely used modified Random Walk estimation was used in the simulation as a secondary reference. In this case, every user has a preferred direction. The distribution of the directions to draw are following normal distribution. The expected value is the preferred direction, variance equals 1, that is the the difference between to adjacent directions. The draw is expected to result a direction that is near the preferred direction, thus the user moves with a drift the points to the preferred direction.

We used these models as a comparison basis to the 7-state Markovian model simulations.

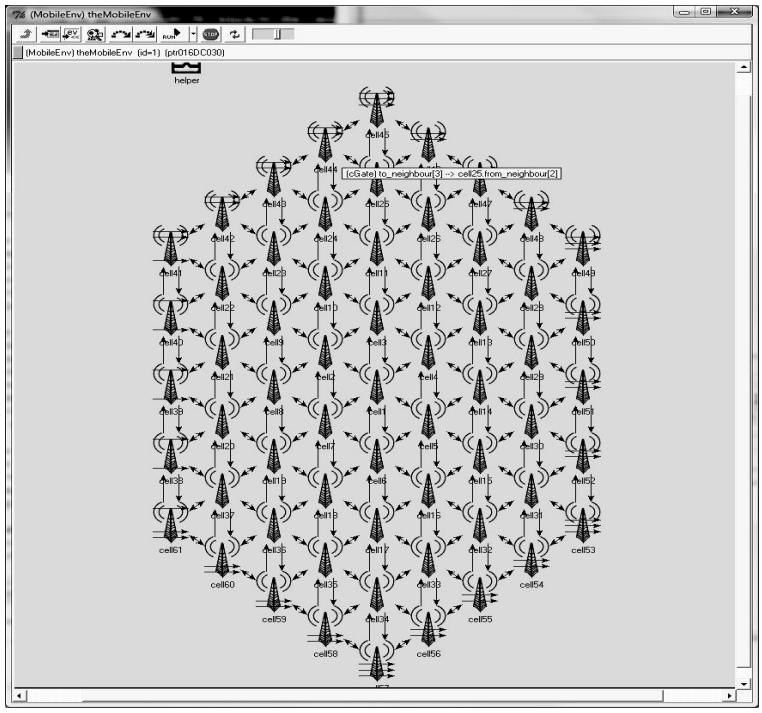


Fig. 25. The mobility simulation environment in OMNet++.

The simulation uses the past and the current trace simulation results to estimate future number of users in each cell and future transition direction of each user. The estimation produces an alternative user distribution in the network. Th estimation error is defined as the difference of the user-distribution in the trace and the model. The quantity of estimation error is interpreted as the measure of accuracy of each mobility model in this paper.

The trace simulation starts earlier than the estimations, during the initial steps the trace reaches its operating stability without cold-start transients. During the warm-up process the trace simulation produces sample data for the estimation procedures which uses the previous trace results as an input to estimate the future user distribution. Each user-transition in the warm-up period is used to derive transition probabilities, motion speed and patterns in the simulation cell-space. These patterns serve as an input for the estimation threads of each mobility model. The models have the same input throughout the simulation process thus the results are comparable.

In the following we discuss the results of two basically different simulation environments. In the first environment the trace simulation follows a strict pattern of forward-backward motion. That is the users are on a self-closing straight path and there is no practical chance of stepping down from this path. In this case the trace simulation works with two directions, forward and backward.

The other environment represents a crowded urbanized area, where each direction has a non-zero probability. Although, the motion is not uniform, each cell has at least one preferred neighbor.

The following plots show the summarized error of the estimations simulated. In the plot-pair, the first shows the forward-backward scenario, while the second plots the result of the urbanized environment.

Figure 26(a,b) shows the sum of the absolute value of error for every cell during the simulation.

Figure 26a) shows that the two Random Walk based models cannot follow the patterns in user fluctuation, the estimation works with a significantly higher error rate than the 7-state Markovian model.

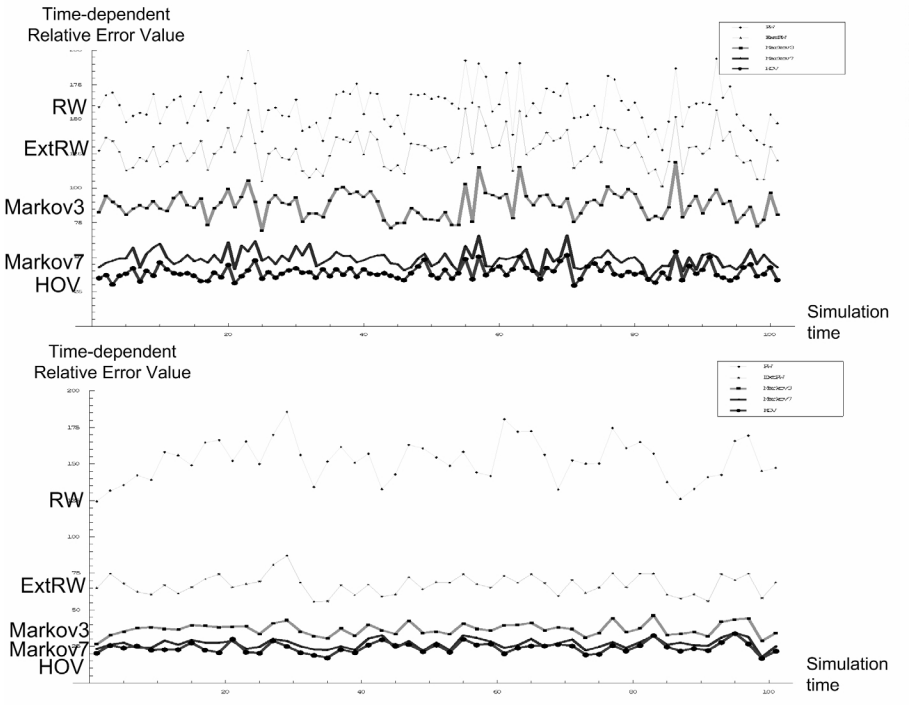


Fig. 26. a. Forward-backward environment error summary; b. Urbanized environment error summary.

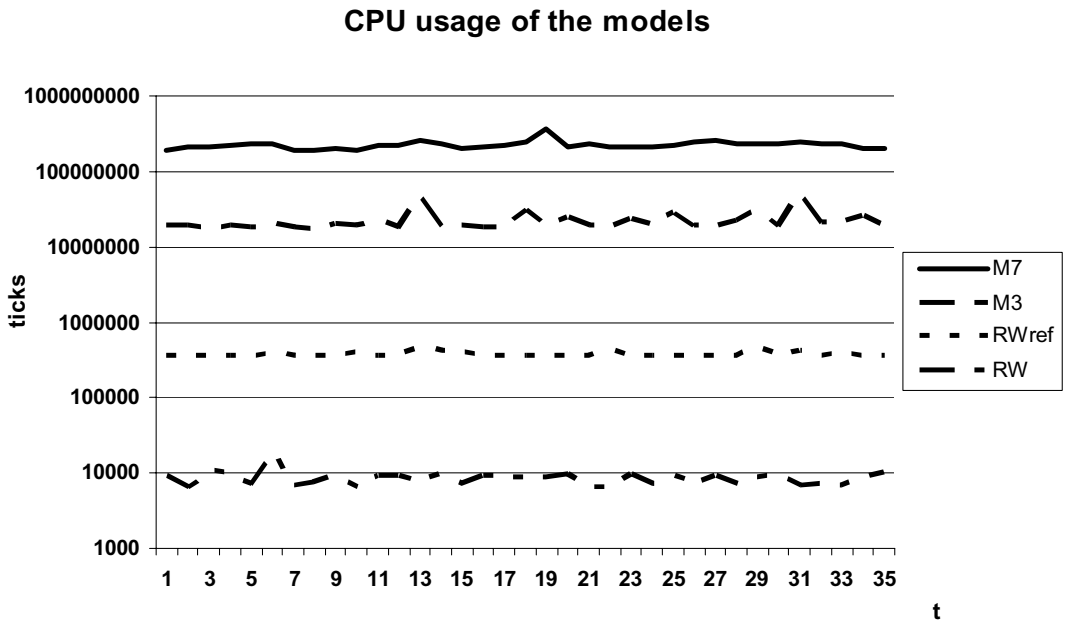


Fig. 27. CPU usage of the models on logarithmical scale.

The 7-state Markovian model has a decreasing error rate ongoing the simulation, along with the Markov chain learning the motion patterns.

Figure 26b) shows a less sharp difference between RW-based and our model, the 7-state Markovian model produces better results. The patterns of the urbanized environment are closer to the simple uniform motion pattern that is used by the Random Walk. This confirms that the RW-like estimations are capable of modeling cases where users are fluctuating randomly without significant or strict patterns.

The relative performance of the models can be seen on the Fig. 27. The execution time of the methods of model are plotted in each timeslot on logarithmical scale.

7. Conclusion

In this paper we proposed mobility models that are capable of simulating real-life user movement accurately. We proposed an alternative Path Based and a Markov-chain based method. The simulation results proved the analytical properties of the proposed mobility models.

The widely used and simple RW model works with a significant error rate at all times in mobility modeling. Since the user movement patterns in the simulation are not completely random due to the streets and geographical circumstances, the uniformly distributed Random Walk pattern cannot model it.

The algorithms that work with memory can estimate with a significantly higher accuracy. That is the 7-state Markovian model produces better results in the prediction of each user position.

We proposed mobility accurate models that are producing better estimations of user distribution in the network, based on present and past network operator data. A network operator may use the 7-state Markov model to make predictions on the future distribution and location of users among radio cells to justify CAC or other QoS decisions.

Acknowledgements

This work was supported by the Mobile Innovation Center, Hungary.

References

[1] A. Jardosh, E.M. Belding-Royer, K.C. Almeroth and S. Suri, Towards Realistic Mobility Models for Mobile Ad hoc Networks, In proceedings of *The Annual International Conference on Mobile Computing and Networking, MobiCom* (September 2003), 217–229.

[2] A. McDonald and T. Znati, Predicting Node Proximity in Ad Hoc Networks: A Least Overhead Adaptive Model for Selecting Stable Routes. In proceedings of *First Annual Workshop on Mobile Ad Hoc Networking and Computing,*(*MobiHoc*), (2000) Boston, pp. 29–33.

[3] A. Fongen, C. Larsen, G. Ghinea, S.J.E. Taylor and T. Serif, Location based mobile computing - A tuplespace perspective, *Mobile Information Systems* **2**(2–3) (2006), 135–149.

[4] A.U. Jayasuriya, *Improved Handover Performance Through Mobility Prediction*, PhD thesis, University of South Australia, 2001.

[5] B. Liang and Z. Haas, Predictive distance-based mobility management for PCS networks, In Proceedings of the *Joint Conference of the IEEE Computer and Communications Societies* (*INFOCOM*) (March 1999), 1377–1384.

[6] C. Abondo and S. Pierre, Dynamic location and forwarding pointers for mobility management, *Mobile Information Systems* **1**(1) (January 2005), 3–24.

[7] D.B. Johnson and D.A. Maltz, Dynamic Source Routing in Ad-Hoc Wireless Networks, *Mobile Computing* **353** (1996), 153–181.

[8] D. Levine et al., A Resource Estimation and Call Admission Algorithm for Wireless Multimedia Networks Using the Shadow Cluster Concept, *IEEE/ACM Transaction On Networking* **5**(1) (February 1997), 1–12.
[9] E. Royer, P.M. Melliar-Smith and L. Moser, An analysis of the optimum node density for ad hoc mobile networks, *In Proceedings of the IEEE International Conference on Communications* (*ICC*) **3** (2001), 857–861.
[10] H. Liao, L. Tie and Z. Du, A Vertical Handover Decision Algorithm Based on Fuzzy Control Theory, *Computer and Computational Sciences, 2006. IMSCCS '06. First International Multi-Symposiums* **2** (24 April 2006), 309–313.
[11] J.J. Garcia-Luna-Aceves and M. Spohn, Source-tree routing in wireless networks, In Proceedings of *The 7th International Conference on Network Protocols* (ICNP), (31 Oct–3 Nov 1999), 273–282.
[12] J. Yoon, B.D. Noble, M. Liu and M. Kim, Building realistic mobility model from coarse-grained wireless trace. In proceedings of the *Fourth International Conference on Mobile Systems, Applications, and Services* (*MobiSys*), Uppsala, Sweden, pp. 177–190, June 2006.
[13] K. Wang and B. Li, Group Mobility and Partition Prediction in Wireless Ad-Hoc Networks, *IEEE International Conference on Communications*, (*ICC 2002*) New York, **2** (April 2002) 1017–1021.
[14] M. Sanchez, Mobilitymodels, from http://www.disca.upv.es/misan/mobmodel.htm (2002, May 30).
[15] M.M. Zonoozi and P. Dassanayake, User mobility modelling and characterization of mobility patterns, *IEEE Journal on Selected Areas in Communications* **15**(7), (1997 September).
[16] *Mathematica Documentation.* Wolfram Research Inc. from http://www.wolfram.com/.
[17] S. Michaelis and C. Wietfeld, Evaluation and comparison of prediction stability for user movement pattern detection algorithms, *European Wireless Conference* (*EW2006*), Athens April, 2006.
[18] R. Chellappa, A. Jennings and N. Shenoy, A comparative study of mobility prediction in fixed wireless networks and mobile ad hoc networks. In proceedings of the *IEEE International Conference on Communications 2003* (*ICC2003*), Alaska, USA, **2** (May 2003), 891–895.
[19] S. Michaelis and C. Wietfeld, Comparison of User Mobility Pattern Prediction Algorithms to increase Handover Trigger Accuracy, *IEEE Vehicular Technology Conference*, Melbourne, **2** (May 2006), 952–956.
[20] T. Camp, J. Boleng and V. Davies, A Survey of Mobility Models for Ad Hoc Network Research, Wireless Communications and Mobile Computing (WCMC), *Special Issue on Mobile Ad Hoc Networking: Research, Trends and Applications* **2**(5) (2002), 483–502.
[21] V. Davies, Evaluating mobility models within an ad hoc network, Master's thesis, Colorado School of Mines, 2000.
[22] V. Simon and S. Imre, A simulated annealing based location area optimization in next generation mobile networks, *Mobile Information Systems* **3**(3–4) (2007), 221–232.
[23] Vincent W.-S. Wong and V.C.M. Leung, Location Management for Next-Generation Personal Communication Network, *IEEE Network* (September/October 2000).
[24] Yuguang Fang Chlamtac and I. Yi-Bing Lin, Portable movement modeling for PCS networks, *IEEE Vehicular Technology* **49** (2000), 1356–1363.
[25] X. Hong, M. Gerla, G. Pei and C. Chiang, A group mobility model for ad hoc wireless networks, In Proceedings of *The ACM International Workshop on Modeling and Simulation of Wireless and Mobile Systems* (*MSWiM*), (1999), pp. 53–60.
[26] X. Hong, T. Kwon, M. Gerla, D. Gu and G. Pei, A Mobility Framework for Ad Hoc Wireless Networks. In proceedings of *ACM 2nd International Conference on Mobile Data Management*, (*MDM 2001*), Hong Kong, (January 2001), 185–196.
[27] Z.J. Haas, A new routing protocol for reconfigurable wireless networks, In Proceedings of *The IEEE International Conference on Universal Personal Communications* (*ICUPC*), (1997), 562–566.

Tamás Szálka was born in Budapest in 1982. He received the M.Sc. degree in Computer Science from Budapest University of Technology and Economics (BUTE) in 2006. Next he started his Ph.D. studies at BUTE, on the Department of Telecommunications. Beside his Ph.D. studies, he is doing engeneering research as Ph.D. student and fellow researcher in the Mobile Innovation Centre of BUTE. As planned, he will obtain his Ph.D. degree in 2011. His work is relatead to the area of mobililty modeling and location based services of cellular networks.

Péter Fülöp was born in Mosonmagyarovar in 1980. He received his M.Sc. degree in Technical Informatics from the Budapest University of Technology and Economics in 2005. Currently he is a PhD student on the faculty of Informatics on the Department of Telecommunication and fellow reserarcher in the Mobile Innovation Centre of BUTE. He is working as a system test engineer at Ericsson Telecommunications Hungary at Research and Development Department. His research interest is IP mobility, interworking of heterogeneous mobile networks and movement modeling in cellular, mobile networks. The PhD thesis is going to be written on Complex Mobility Management Systems and Applications.

Sándor Imre was born in Budapest in 1969. He received the M.Sc. degree in Electronic Engineering from the Budapest University of Technology (BUTE) in 1993. Next he started his Ph. D. studies at BUTE and obtained dr. univ. degree in 1996, Ph.D. degree in 1999 and DSc degree in 2007. Currently he is carrying his teaching activities as Head of the Dept. of

Telecommunications of BUTE. He was invited to join the Mobile Innovation Centre of BUTE as R&D director in 2005. His research interests include mobile and wireless systems. Especially he has contributions on different wireless access technologies, mobility protocols and reconfigutable systems.

Sándor Szabó was born in 1977. He received the M.Sc. degree in Electronic Engineering from the Budapest University of Technology (BUTE) in 2000. Next he started his Ph. D. studies at BUTE. Currently he is carrying his teaching activities as an assistant professor at the Dept. of Telecommunications of BUTE. He leads research projects, and also a project leader at the Mobile Innovation Centre of BUTE. His research interests include mobile and wireless systems, mobility modeling, mobility management and the integration of heterogeneous wired and wireless networks.