Learning movement patterns in mobile networks: a generic method.

Jean-Marc François, Guy Leduc and Sylvain Martin[1]

Research Unit in Networking (RUN)
Department of Electrical Engineering and Computer Science
Université de Liège, Belgium
{**francois,leduc,martin**}**@run.montefiore.ulg.ac.be**

***Abstract*: Predicting terminals movements in mobile networks is useful for more than one reason, in particular for routing management. A way to do such prediction is to learn the movement patterns of mobile nodes passing by an access router. In this paper, the information (e.g. layer 2 measurements) related to the different paths followed by mobiles are learned using a hidden Markov model. Simulations have been done using this method and show it can handle different layer 2 signals and collect statistical information when no such signal is available.**

1. Introduction

In mobile networks, *mobile hosts* (MHs) are allowed to move without losing their connectivity: either they have a wireless interface talking to the nearest *access point* (AP), or they are simply plugged and unplugged at a new location. The first router linking MHs to the wired network is called the *access router* (AR). In this context, the movement prediction problem is guessing the next access router(s) a MH will be linked to.

This prediction can be particularly useful to assure a given level of QoS despite the typically large jitter and error rates of wireless networks. Every *handover* (leaving an AP to reach the next) leads to packet losses and requests a registration (*e.g.* [9]) that might induce a long delay. These shortcomings can be reduced thanks to micro-mobility protocols ([4, 15, 2, 17]), but can't be eliminated.

The mobility prediction methods aim to anticipate ressource allocation and allow proactive registrations.

In [6], the top of each MN's web cache is sent to the most probable next ARs guessed using a *learning automaton*. The same learning method is used in [7] so as to stochastically reallocate TCP datagrams to neighboring ARs, thus improving TCP performance. [5] introduced a statistical method to correlate MNs' moving patterns characteristics (time of the day, previous AR, time elapsed in the cell,...) with their trajectories; this information if used to keep the handover dropping probability under a chosen threshold.

Some information can be sent by the MHs themselves to guess their movement. [16] studies the performance of a method based on GPS and maps to learn where and when the handovers usually occur. Liu *et al.* ([12]) uses each MH's knowledge of its typical movements and signal strength measurements (*via* a self-adaptive extended Kalman filter and assumptions about ATM cells geometry) to guess its path.

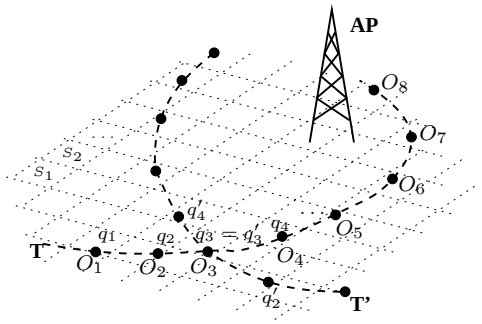

Figure 1: Two trajectories near an AP and the associated set of observations and states.

Mobility prediction is important in the context of ad hoc networks (*i.e* with no fixed infrastructure) where it's used to estimate route stability ([11]).

As one can see, different kinds of methods can be applied; some of them rely on statistical measurements, others on information emitted (or learned) by the MHs.

To be as generic as possible, the movement patterns learning process presented here tries to accomodate any type of information emitted by MHs, and gives a simple statistical result when there is none. It is based on a simple AI technique and gives to mobiles a simple role.

It should be noticed that the method presented hereafter is, in a way, complementary to the various papers talking about mobility models ([8, 1, 3]): whereas these works set mobiles' motion behaviours *a priori*, those behaviours are here taken from observed terminals' movements and directly used for prediction purposes.

The next section shows how mobiles' motion can be expressed in terms of Markov processes and section 3. defines a way to model them. Section 4. explains how path prediction can be done using these models and gives the results of some simulations. Section 5. gives some extensions that can easily be added to the method. The last section concludes the paper.

2. Movement patterns and Markov processes

Let's consider somebody walking down a street using a mobile device connected to a wireless IP network. This device communicates with a given *Access Point* (AP) and can regularly measure some information directly related to the path it follows (*e.g.* given by the layer 2

[1] Sylvain Martin is a Research Fellow of the Belgian National Fund for Scientific Research (FNRS).

This work has been partially supported by the Walloon Region in the framework of the WDU programme (ARTHUR project), and by the Belgian Federal Office for Scientific, Technical and Cultural Affairs (SSTC) in the framework of the IAP programme (MOTION P5/11 project).

protocol or a separate apparatus such as a GPS).

Reporting these measurements at discrete time steps $t = 1, 2, 3, \ldots$, we get a sequence of observations (respectively $O_1, O_2, O_3, \ldots$), each of them being a vector of discrete or continuous values.

Since we are interested in the paths followed by mobiles, let's divide the zone near an AP in small areas (or *states*) Q_i and try to match those states with sequences' observations. Let q_t be the state matching the tth observation (see figure 1).

If mobiles' movements were memoryless, *i.e.* if:

$$\forall t : P[q_t = Q_i \mid q_{t-1} = Q_j, q_{t-2} = Q_k, \ldots] = P[q_t = Q_i \mid q_{t-1} = Q_j] \quad (1)$$

then it would be a first order Markov chain and the model shown hereafter would be particularly well suited. Unfortunately, the simple example at figure 1 shows that (1) doesn't hold (one can't distinguish between trajectories T and T' knowing q_3 or q'_3 only, but can if it's given q_2 or q'_2). Despite that, simulations under different conditions show that it gives good results if it's used as explained later on (see section 4.1.).

Considering a walker without any a priori information linked to an AP for a given time, we can get an observation sequence S. S is stochastic for at least two reasons:

a. each observation can be corrupted by noise or some other kind of measurement error.

b. nobody walks randomly, but usually several paths could be followed in an AP's zone of influence.

Because of a., the state-to-observation matching can only be probabilistic, so we'll define a *observation's probability function* of a given state. Because of b., guessing q_t knowing q_{t-1} can't be done exactly, so we'll define a *state transition probability matrix*.

In practice, the different states q_i a mobile is passing by are hidden, but can be deduced from observation sequences. This kind of (doubly stochastic) process can be modelled using *Hidden Markov Models* (HMMs).

3. Markov processes and Hidden Markov Models

A HMM ([14, 13]) is a model of observation sequences $S = O_1, O_2, O_3, \ldots$ as explained in section 2.. It is characterized by:

- N *states*: $Q_1, Q_2, \ldots, Q_N$. The state at time t is denoted q_t;
- *a set of observation's probability distibution functions*: $B = \{b_i(O)\}$.

$$b_i(O) = \mathrm{P}[O \text{ at } t \mid q_t = Q_i] \quad (2)$$

  $(\forall i = 1, 2, \ldots, N)$.

- *a state transition probability matrix*: $A = \{a_{ij}\}$.

$$a_{ij} = \mathrm{P}[q_t = Q_j \mid q_{t-1} = Q_i] \quad (\forall i, j = 1, 2, \ldots, N) \quad (3)$$

  The following properties hold:

$$0 \leq a_{ij} \leq 1 \qquad (\forall i, j = 1, 2, \ldots, N) \quad (4)$$

$$\sum_{j=1}^{N} a_{ij} = 1 \qquad (\forall i = 1, 2, \ldots, N) \quad (5)$$

- *the probability that Q_i is the initial state of a sequence*:

$$\pi_i = \mathrm{P}[q_1 = Q_i] \qquad (\forall i = 1, 2, \ldots, N) \quad (6)$$

  Let π be the set of these probabilities: $\pi = \{\pi_i\}$.

A HMM model is usually denoted $\lambda = (A, B, \pi)$ which reminds of the parameters involved.

The probability of observing a sequence S and a state sequence $Q = q_1, q_2, q_3, \ldots q_T$ given a model λ is simply the probability that q_1 is the initial state, times the probability of observing O_1 at q_1 (hereafter denoted $\pi_{q_1} = \pi_i$ with i such that $q_1 = Q_i$), times the probability of going from q_1 to q_2,...:

$$\mathrm{P}[S \mid Q, \lambda]\ \mathrm{P}[Q \mid \lambda] = \pi_{q_1} b(O_1) a_{q_1 q_2} b(O_2) a_{q_2 q_3} \ldots a_{q_{T-1} q_T} b(O_T) \quad (7)$$

To get the probability of S given λ, we simply sum the probability of all the possible state sequences:

$$\mathrm{P}[S \mid \lambda] = \sum_{\text{all } Q} \mathrm{P}[S \mid Q, \lambda]\ \mathrm{P}[Q \mid \lambda] \quad (8)$$

Another problem of great interest is: how can one find the most probable state sequence matching a given observation sequence S:

$$Q^* = \operatorname*{argmax}_{Q}\ \mathrm{P}[S, Q \mid \lambda] \quad (9)$$

The calculation of (8) and (9) are well known problems that have been resolved efficiently ([13]).

The last unresolved question is: how can one adjust the model parameters of $\lambda = (A, B, \pi)$ to best suit a set of observation sequences ? It depends on what *best suits* means in this context. The two most used criteria are the *maximum likelihood* (optimizing $\mathrm{P}[S \mid \lambda]$ — see [13]) and the *state optimized likelihood* (optimizing $\mathrm{P}[S, Q^* \mid \lambda]$, where Q^* is given by (9) — see [10]), which lead to classical iterative procedures.

4. HMMs and path prediction

4.1. Overview

In mobile IP networks, access routers (AR) manage mobile nodes (MN) arrivals and departures. In this paper, the movement pattern learning takes place in each AR and MN's role is kept minimum.

In this context, the path prediction's purpose is twofold:

- learning the typical MN's movement patterns so as to guess their future AR;
- sending to each neighbouring AR statistical information related to MNs that are likely to turn towards it (possibly together with related indication, as the needed QoS).

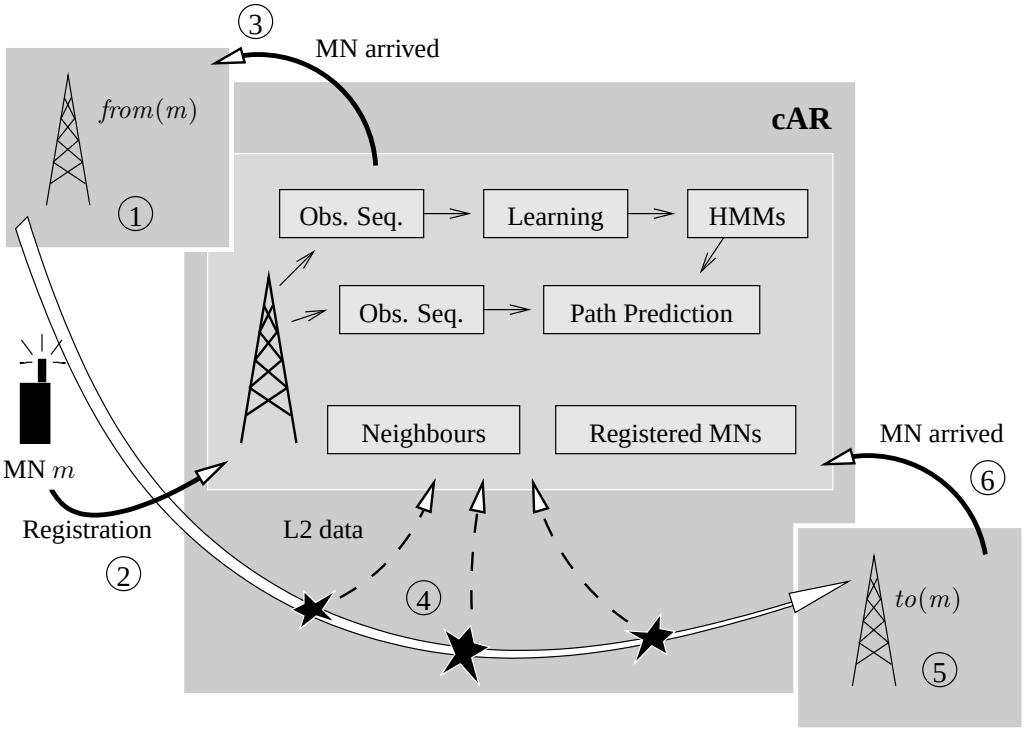

Figure 2: **Overview.** Once cAR has seen enough MN's passing by in order to learn their typical behaviour, the path prediction procedure happens as follow : (1) A MN m leaves the AR $from(m)$ and goes towards cAR. —(2) It registers with cAR. —(3) cAR informs $from(m)$ about m's arrival. It adds $from(m)$ to its neighbours list (if it has not been done before, which is unlikely since we suppose that the learning process is terminated). —(4) From time to time, m emits observations about its journey; this lets cAR build a sequence and check its probability using the HMMs. When it appears that one HMM is clearly more probable than the others, the matching neighbour can be informed. —(5) The same procedure takes place when m leaves cAR and reaches $to(m)$. —(6) $to(m)$ informs cAR about m's arrival (just as cAR did).

We say AR_j is *in the neighbourhood of* AR_i if mobile nodes regularly travel from AR_i to AR_j; therefore, the neighbouring relation is dynamic and must be constantly reevaluated. To let an AR know its neighbours, we suppose that every MN remembers the last AR he was registered with; this way, a neighbour can give an AR notice of a mobile's arrival.

This neighbouring relation is a key point here, and it is determined by the way the mobiles travel between the ARs. Let R be the set of MNs registered with the current AR *cAR* (*i.e.* the one we are interested in). We denote $from(m)$ the AR a MN m visited just before the current one; similarly, $to(m)$ is the next AR m will register with. The current AR's set of neighbours is denoted N; thus, $\forall m \in R : from(m) \in N$ and $to(m) \in N$.

It could be possible to catch the behaviours of all the MNs an AR is involved with using only one HMM, but we don't think this is the best way to organize the learning process. In order to get observation sequences as close as possible to (1), one shouldn't mix unsimilar sequences. It is thus proposed to create a different model for sequences built by mobiles which are not going to or coming from the same AR; so, the number of HMMs grows as the number of neighbours squared. Let $\lambda_{i,j}$ be the HMM associated with the mobiles going from AR_i to AR_j. We will see that the number of sequences used to build $\lambda_{i,j}$ will take some importance; let $p_{i,j}$ be that number and p_i their sum ($\sum_{j \in N} p_{i,j}$).

Figure 2 gives an overview of the different actions involved when a MN m is passing by the access routers $from(m)$, cAR and finally $to(m)$. On its travel, it can send L2 information (*e.g.* piggybacked with other kinds of data) which are received and stored by cAR. When registering with $to(m)$, the MN sends the address of the AR he just left (cAR) together with its Home Address; then, $to(m)$ informs cAR about the MN's arrival, so that it can:

- add $to(m)$ to its neighbours list. This list is regularly cleaned up so as to remove unrefreshed entries;
- match the MN's home address with the addresses of recently-gone MNs and find the associated (L2) observation sequence;
- use it to build $\lambda_{i,j}$. It cannot be done before enough sequences have been collected.

Once the learning process is stabilized, the path prediction itself takes place. A MN m generates a sequence $S = O_1, O_2, O_3, \ldots$; let S_i be S truncated so that only the first i observations remain. Each time an new observation is emitted, the AR can compute

$$P_j = \text{P}[S_i \mid \lambda_{from(m),j}] \quad \forall j \in N$$

using (8). The ratio between those probabilities weighted by $p_{from(m),j}/p_{from(m)}$ tells us towards which

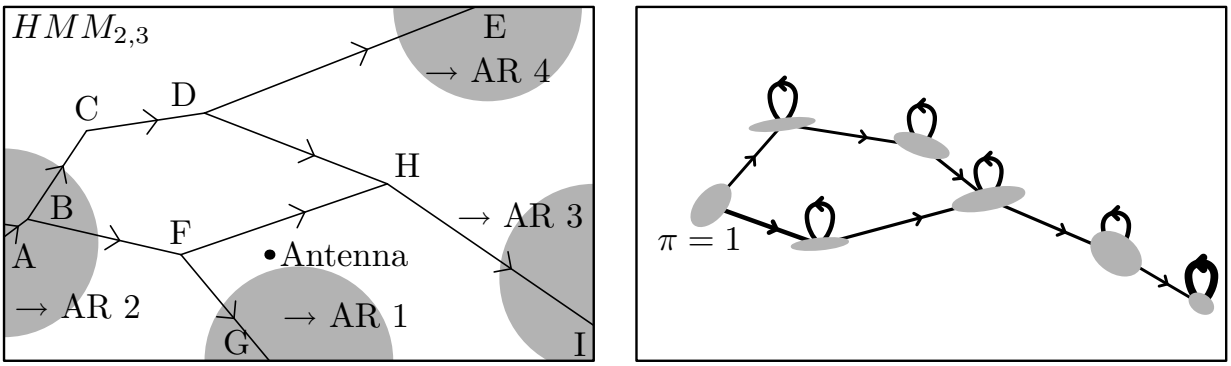

Figure 3: **A simulation of path learning.** The left picture shows a map of one-way roads. The cars that are coming from A and find their way to E, G and I at random. As they are passing by, they generate sequences of observations; those sequences can be learned by a HMM, as shown in the right figure. Each state is drawn using its gaussian observation probability function's covariance ellipse (enlarged 1.5 times for clarity).

neighbour m is likely to go, and the certainty of doing so.

Let's consider the issues that could prevent the method from scaling. The only messages needed to apply the method are shown in figure 2; the amount of the data sent during the registration process is small and the various data sent during the MN journey can be piggybacked. The computation of the HMMs given in the next section takes less than 1 second per model[2]; computing the probability of a sequence given a model can be done in a few milliseconds[3].

4.2. Simulations

Figure 3 shows the results given by a simulation of the above procedure. Cars equipped with a GPS are passing by an AR; thus, the observations are 2-dimensional (x, y) vectors giving the position of the car. To be as generic as possible, we suppose that all the observations probabilities involved in this paper can be modeled using a gaussian probability function; thus, results could be made even better if probability functions were built more carefully, at the expense of generality.

The left picture shows the topology of some one-way roads. The vehicules come from A and travel as indicated by the arrows; at each crossing, the cars choose their way randomly. The speed of each car is a random constant (linearly distributed between a minimum and a maximum value) and observations are emitted regularly. We can be given an insight into the computed HMM using a graphical representation (see the right picture): the transition probabilities are depicted by lines of different widths, and the probability distribution of each state by a covariance ellipse[4].

As one can see in figure 4, for the example given, the HMM easily discriminates the different sequences and the next-neighbour prediction should never fail.

[2]Computation done with 400 sequences using a Java prototype on a Pentium 1.2GHz.

[3]Using the same prototype. The complexity of the algorithm is N^2T, N being the number of model's states and T the length of the sequence.

[4]The *covariance ellipse* (or *error ellipse*) of a bivariate gaussian probability function is an ellipse of constant probability density, centered at each variable's mean, and such that the probability of an observation to be in this ellipse is ≈ 0.39.

Now, an important point is: how can we replace the GPS data with a different one, for example the mobile-antenna distance (which could be derived from the received antenna power) ? In fact, nearly no modifications is needed at all: the HMMs just learn 1-dimensional vectors instead of 2-dimensional ones. Figure 5 shows the results with this new configuration; they should be compared with those of figure 4.

Simulations of this technique have been realized (using mobiles-antenna distances) with several maps of different complexity. With simple maps, such as the one presented at figure 3, the prediction gives nearly perfect results (between 98 and 100%). A more complex map made of 40 intricated roads and 7 potential neighbouring ARs shows the importance of the sequences length and HMM size: 5 states HMMs modeling sequences of 5 observations on average guess the correct next neighbour 67% of the time. This number is as high as 82% when using 10 states HMMs with sequences of 10 observations on average. Interestingly, when counting as a good guess the actual next AR being one of the first two most probable next neighbours, those figures grow to 88% and 94% respectively. Mixing short sequences with large HMMs (or the other way round, long sequences with small HMMs) gives intermediate results. The influence of noise can be reduced using longer sequences, up to a certain level where the sequences become indistinguishable.

5. Extensions

One could argue that more information about mobiles displacements are needed than just the different next-neighbour probabilities. For example, a potential problem is to estimate when a handover will occur so as to prepare it ahead of schedule, but not too much.

Another problem is how to handle mobiles with no means of generating any observation at all. In this case, we must resign to do a statistical estimation of the mobiles movements (an example of how useful this information is can be found in [5]).

Those problems are studied in the next sections.

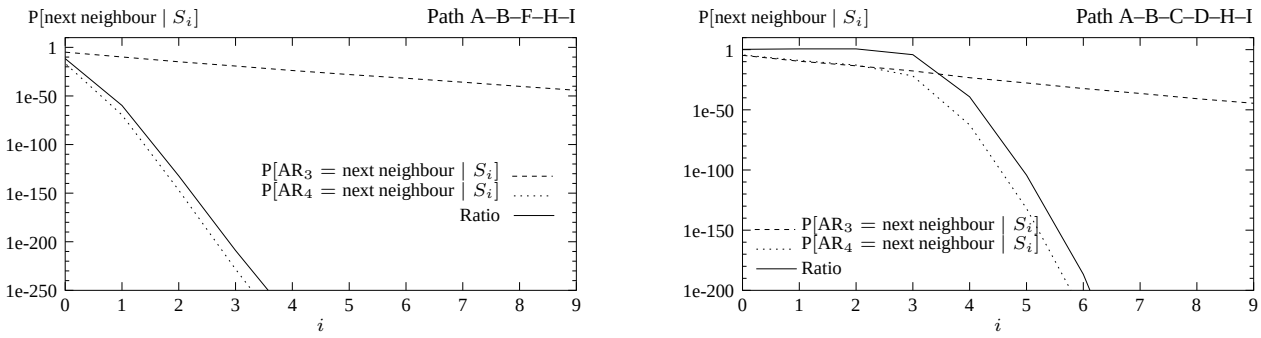

Figure 4: **Prediction (using GPS coordinates).** Those graphs show the evolution of $\mathrm{P}[S_i \mid \lambda]$ with i (*i.e.* how the next-neighbour predition evolves as the time passes) for a mobile that goes from A to I (see figure 3). The two dashed curves correspond to the probability that the sequence belongs to each of the two given HMM; the plain one is their (and should be smaller than 1 if the prediction is right). The left-hand graph shows that when a mobile chooses to reach I *via* B, F and H, the prediction is easy because this path is very different from the one that reaches E (A-B-C-D-E). On the contrary, the right-hand graph shows that while the mobile travels towards E, the probabilities ratio stays close to 1 during the first 3 observations.

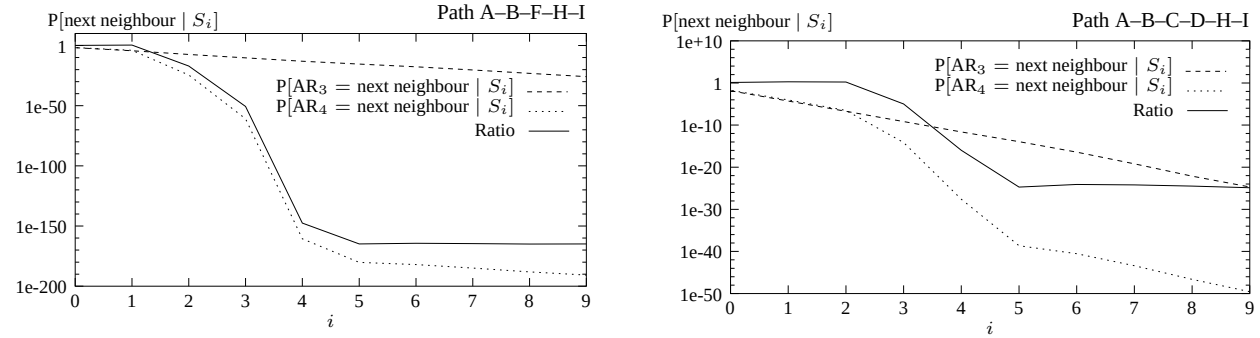

Figure 5: **Prediction (using antenna distance)**. Those graphs are very similar to those of figure 4. Here, the observations are the distances between the mobile and the antenna it is linked to (this antenna is situated near the middle of the road map; see figure 3). The prediction process behaves much like with a GPS, but is less precise.

5.1. Learning more

This section describes how to take advantage of the MNs' path learning to learn other kinds of information. For example, suppose one wants to match an observations sequence S (of length i) with the expected amount of time before the next handover. The solution is straightforward: apply the same procedure as we've just seen, and, before they enter the HMM learning process, add a dimension (the time we are interested in) to each observation vector. Thus, if we consider that the observations are emitted at a constant rate, this conversion can be written:

$$S = \begin{pmatrix} o_1^1 \\ \vdots \\ o_d^1 \end{pmatrix} \begin{pmatrix} o_1^2 \\ \vdots \\ o_d^2 \end{pmatrix} \cdots \begin{pmatrix} o_1^i \\ \vdots \\ o_d^i \end{pmatrix} \rightarrow$$

$$S' = \begin{pmatrix} o_1^1 \\ \vdots \\ o_d^1 \\ t \end{pmatrix} \begin{pmatrix} o_1^2 \\ \vdots \\ o_d^2 \\ \frac{i-2}{i-1}t \end{pmatrix} \cdots \begin{pmatrix} o_1^j \\ \vdots \\ o_d^j \\ \frac{i-j}{i-1}t \end{pmatrix} \cdots \begin{pmatrix} o_1^i \\ \vdots \\ o_d^i \\ 0 \end{pmatrix}$$

... where t is the time taken to generate S (from registration to handover). After that, one can use (9) to guess the most probable state a given sequence ends with, and see the associated time distribution.

Figure 6 shows the results given by this method. Tests have been produced with two HMM models: $\lambda_{2,3}$ with 7 states (as in fig. 3) and with 15 states. The t axis reflects the exact time-to-go when the observation is emitted; μ_t and σ_t are respectively its approximated value and standard deviation computed with the HMMs. As one could expect, increasing the model's complexity leads to more accurate time predictions and smaller deviations.

5.2. Statistical prediction

If we aim at building a prediction mechanism as generic as possible, we must look at the important case of mobiles unable to generate any kind of hint about their movements.

First of all, considering a MN m, notice that $\mathrm{P}[to(m) = \mathrm{AR}_j \mid from(m) = \mathrm{AR}_i] = p_{i,j}/p_i$ holds and that the sum of those probabilities $\forall m \in R$ gives the expected proportion that will reach AR_j. Clearly, we would get a more precise result if we could take into account for how much time the MNs is registered.

We can consider the situation depicted here as a particular case of the section 5.1. above, with sequences S_i containing an observation vector of dimension 0. As before, we add the time elapsed between the mobile registration and its handover to this null observation; simple single-state HMMs are sufficient to learn those degenerated sequences.

Once the learning process has taken place, one can

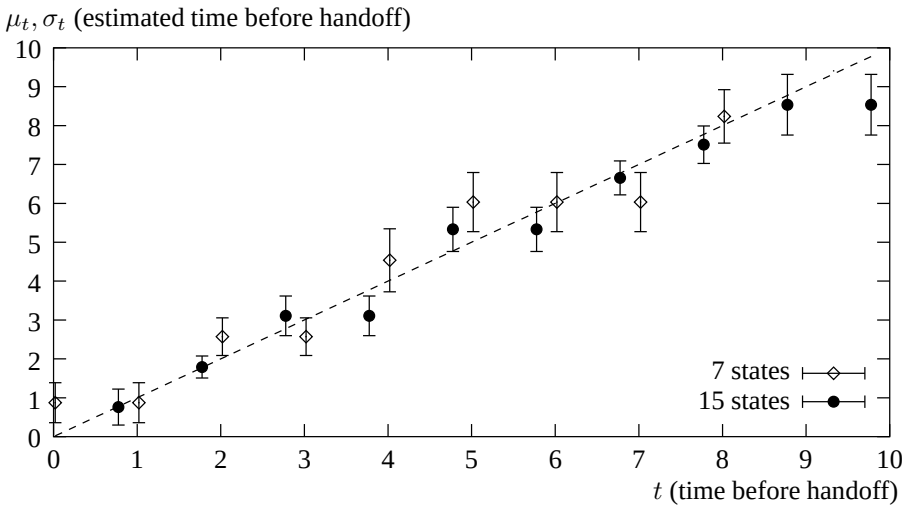

Figure 6: **Departure time prediction.** Two sequences have been generated by two MNs travelling on the same road, at the same speed. The graph shows the time-to-go predicted by two HMMs with different number of states. The computed standard deviations are given by the bars above and underneath the dots.

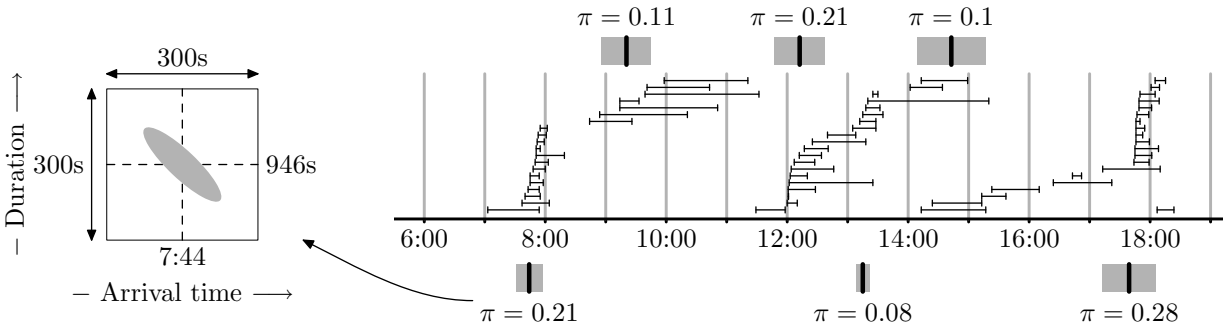

Figure 7: **Departure prediction.** Each segment above the timeline shows when a MN arrived and how long he stayed. When feeding a HMM learning algorithm with this information, one gets a set of states depicted using grey rectangles crossed by a line; the line (resp. grey zone) represents the mean arrival hour (resp. arrival hour's standard deviation) of its observation probability function (for a mean registration-to-handover delay). The left graph shows the covariance ellipse of the most probable state for observations emitted at about 8:00.

use the probability function $b_{from(m),j}$ associated with $\lambda_{from(m),j}$'s single state to compute the probability a given MN m has to do a handover to reach AR_j in a timeslot of T time units if it arrived t_m time units ago (denoted $p_j^{m,T}$; the sum of these values yields the expected number of mobiles that will travel to AR_j in the next timeslot of T time units (denoted n_j).

$$p_j^{m,T} = \frac{p_{from(m),j}}{p_{from(m)}} \frac{\int_{t_m}^{t_m+T} b_{from(m),j}(x)\,\mathrm{d}x}{\int_{t_m}^{+\infty} b_{from(m),j}(x)\,\mathrm{d}x}$$

$$n_j = \sum_{\forall m \in R} p_j^{m,T}$$

If one needs to predict the bandwith taken by MNs going to a particular neighbour, then using 2-dimensional vectors (holding the registration-to-handover delay and the bandwith used) is well suited. The covariance of those vectors' gaussian model could point out, for example, that fast MNs usually use more bandwidth (*e.g.* because high speed trains give wireless multimedia services).

In a wired environment where MNs are, for example, laptops liable to leave the network at any time, HMMs could be used to predict departure hours using the learned visiting habits. With that aim, the sequences can be made of vectors like this:

$$\begin{pmatrix} \text{registration-to-handover delay} \\ \text{registration time of day} \end{pmatrix}$$

Figure 7 shows the results given by this method; it depicts what might be the visits to a students' Internet room in a campus. One can see that states are regularly distributed over the day, allowing us to discover some typical movement behaviour. The left-hand "8 o'clock probability function" illustrates this point: it appears that, just before 8 (beginning of the first course), several short visits are made (*e.g.* to get Emails) and that the later they begin, the shorter they are (so as not to be late).

6. Conclusions and future work

In this paper, the choice has been made not to rely on the MNs to know their typical path, but rather to let the ARs do the learning process. This choice is meant to save MNs' computation time and/or bandwith on wireless channels.

An overview of how to apply a well-known learning

method (*i.e.* Hidden Markov Models) to the path prediction problem in mobile networks has been given.

It's been shown that HMMs give good results, yield to a fairly generic method, and can be extended to cope with typical problems involved in this context. In particular, the special case of MNs in wired networks has been studied.

It would be interesting to study the performance of this method when dealing with various mobility models; the prediction should become less accurate when dealing with behaviours of growing randomness.

In the future, tests with real-life data should be undertaken, so as to see the differences with the simulations proposed here.

Acknowledgements

Thanks are given to Pierre Geurts for his help, explanations and remarks.

REFERENCES

[1] C. Bettstetter. Smooth is better than sharp: a random mobility model for simulation of wireless networks. In *Proceedings of ACM Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 19–27, Rome, Italy, July 2001.

[2] A. T. Cambell, J. Gomez, et al. Cellular IP. Internet draft, Juanuary 2000. draft-ietf-mobileip-cellularip-00.txt.

[3] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for Ad Hoc network research. *WCMC: Special Issue on Mobile ad hoc Networking: Research, Trends and Applications*, 2(5):483–502.

[4] Claude Castelluccia. A hierarchical mobile IPv6 proposal. Technical report, INRIA TR-226, November 1998. http://www.inrialpes.fr/Planete/people/ccastel/.

[5] Sunghyun Choi and Kang G. Shin. Predictive and adaptive bandwidth reservation for hand-offs in QoS-sensitive cellular networks. *ACM SIGCOMM Computer Communication Review*, 28(4):155–166, 1998.

[6] Stathes Hadjiefthymiades and Lazaros Merakos. ESW4: enhanced scheme for WWW computing in wireless communication environments. *ACM SIGCOMM Computer Communication Review*, 29(5):24–35, 1999.

[7] Stathes Hadjiefthymiades and Lazaros Merakos. Using path prediction to improve TCP performance in wireless/mobile communications. *IEEE Communications Magazine*, (40-8):54–61, August 2002.

[8] Xiaoyan Hong, Mario Gerla, Guangyu Pei, and Ching-Chuan Chiang. A group mobility model for Ad Hoc wireless networks. In *Proceedings of MSWiM'99*, Seattle, WA, August 1999.

[9] D. Johnson, C. Perkins, et al. Mobility support in IPv6. Internet draft, February 2003. draft-ietf-mobileip-ipv6-21.txt.

[10] B. H. Juang and L. H. Rabiner. The segmental k-means algorithm for estimating the parameters of hidden markov models. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 38(9):1639–1641, September 1990.

[11] Sung-Ju Lee, William Su, and Mario Gerla. Wireless ad hoc multicast routing with mobility prediction. *Mobile Networks and Applications*, 6(4):351–360, August 2001.

[12] Tong Liu, P. Bahl, and I. Chlamtac. Mobility modeling, location tracking, and trajectory prediction in wireless ATM networks. *IEEE JSAC*, 16(6):922–936, August 1998.

[13] L. R. Rabiner. An tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–282, February 1989.

[14] L. R. Rabiner and B. H. Juang. An introduction to hidden markov models. *IEEE ASSP Magazine*, 34-3:4–16, June 1986.

[15] R. Ramjee, T. La Porta, S. Thuel, et al. IP micromobility support using HAWAII. Internet draft, July 2000. draft-ietf-mobileip-hawaii-01.txt.

[16] Wee-Seng Soh and Hyong S. Kim. QoS provisioning in cellular networks based on mobility prediction techniques. *IEEE Communications Magazine*, (41-1):86–92, January 2003.

[17] András G. Valkó. Cellular IP: A new approach to Internet host mobility. *Computer Communication Review*, 29(1):50–65, January 1999.