

Density Estimation for Out-of-Range Events on Personal Mobile Devices

Arjan Peddemors^{**}, Henk Eertink^{*} and Ignas Niemegeers^{*}

^{*}INCA Group, Telematica Instituut
The Netherlands

{Arjan.Peddemors, Henk.Eertink}@telin.nl

^{*}WMC Group, EWI Faculty, TU Delft
The Netherlands

I.Niemegeers@ewi.tudelft.nl

ABSTRACT

Over the years, personal mobile devices have obtained increasing capabilities to concurrently connect to surrounding networks and nearby other devices. Good knowledge on the dynamics in the availability of these heterogeneous entities constitutes essential input for various data communication problems, ranging from the adaptation of applications on mobile devices to multi-homed situations, to the optimization of routing protocols for delay tolerant networks. In this paper, we focus on a method for the *prediction in time* of the loss in visibility of currently in-range network entities, as observed on a personal mobile device. We are interested in estimating the full probability density function of the time of these out-of-range events, as this allows us to ask arbitrary questions such as: what is the probability of losing connection X in the next Y minutes? To do so, we model the mobility of the user by applying kernel density estimation on previously observed mobility traces collected during a user experiment we ran with 12 participants in a six week period, logging cellular, 802.11 wireless LAN, Bluetooth, and various other events on the participant's mobile device.

Categories and Subject Descriptors

C.2.1 [Computer-Communication Networks]: Network Architecture and Design – Wireless communication; I.6.5 [Simulation and Modeling]: Model Development – Modeling methodologies

General Terms

Algorithms, Measurements, Design

Keywords

Mobility Prediction, Wireless Tracing, Delay Tolerant Applications, Kernel Density Estimation

1. INTRODUCTION

On mobile devices with multiple network interfaces, applications often can obtain network connectivity in a concurrent fashion. As the user moves around, the availability and characteristics of

network resources on these devices – especially those based on short range wireless links – change over time. Delay tolerant applications can improve their performance by taking into account the dynamics in the state of network resources as previously observed for a particular user on a particular device. For example, a lengthy synchronization operation may be postponed if the necessary network resource is expected to be out-of-range on short notice. Or, a download may be deferred if the probability is high that a high-bandwidth network becomes available soon.

Predicting the next availability in time of wireless networks or individual network entities – such as base stations, access points and other types of nodes – consists of *assigning probabilities to future points in time* at which these entities get in-range and get out-of-range for the first time after the current time. We focus on predicting out-of-range events for network entities that are already in range, under the assumption that the same approach, perhaps in a more elaborate form, can be applied to the prediction of in-range events. The duration of the availability of network entities may vary multiple orders of magnitude, depending on the mobility of the user and the range of network technologies taken into account. Our approach must therefore be able to deal with these widely varying durations.

A common way to address this prediction problem is to describe the dynamics in network visibility with a Markov model. Some of the previous work looking at related problems [14] is based on this approach, using straightforward or derived forms of Markov models, where the states in these models represent the states in visibility of surrounding networks. It can be highly useful to predict the attachment to the next network entity, to facilitate, for instance, fast handovers during the process of mobility management. In such a case, the order of state transitions is more relevant than the precise timing of these transitions. However, when timing *is* relevant, such as in our case, straightforward Markov models do not deliver easy solutions. We want to obtain the probability distribution for a future time that another state becomes active for the first time after the current state. This is a classical problem, known as the first passage time problem. It is in general difficult to solve analytically, even for simple Markov models with time-homogeneous state transitions, and requires exponential time to compute recursively (see [6], pp. 25) for discrete time models. Another drawback of Markov models is the following. When assuming the Markov property for state transitions as well as assuming time-homogeneous transitions – as is often the case, the waiting time distribution in the current state can only have an exponential shape, because this is the only distribution that is memoryless. When observing the mobility of

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MobilityModels '08, May 26, 2008, Hong Kong SAR, China.

Copyright 2008 ACM 978-1-60558-111-8/08/05...\$5.00.

real users, this is clearly not the case: the probability to leave the current state, i.e. detach from the current network entity, may vary considerably during the day for real users and thus does not follow an exponential shape. Obviously, this may be fixed by using more elaborate Markovian models or drop the time-homogeneous assumption, to the expense of more complexity. For instance, the mobility model in [8] is based on a semi-Markov process allowing for arbitrarily distributed detach times. In all, these considerations were an important motivation for us to look for an alternative approach.

In this paper, we propose to determine the probability density function of the time of a future event straight from the available data. So, if we have a set of samples describing previous begin times and end times of visibility intervals of a network entity, we use this set to predict the future out-of-range time of this entity (when it currently is visible). Although conceptually simple, we need to address a number of issues that arise when looking at this approach in more detail. We argue for the usage of *kernel density estimation* to compute the probability density function and compare different models for prediction, based on the absolute time-of-day of the interval end time, the duration of the interval and the absolute end-time conditional on the begin time. For the conditional prediction, we use the method of conditional kernel density estimation.

To evaluate the out-of-range prediction, we take the data of a user experiment we ran with 12 participants in a six week period with an accumulated monitoring time of more than one year. The experiment logs the cellular, 802.11 wireless LAN, Bluetooth and (indirectly) fixed USB status events as well as the IP configuration over established active links, such as an active GPRS PDP context or an active 802.11 association. These traces are used to discuss a prediction example that identifies occasions where the probability is high (> 0.9) that a network entity remains visible for the next hour.

The rest of the paper is organized as follows. Section 2 introduces the definitions we use and our approach. Section 3 describes the out-of-range prediction using kernel density estimation. Section 4 describes the user experiment, and section 5 provides a prediction example using the real-world data from the experiment. Section 6 gives a short overview of related work and we wrap up with conclusions in section 7.

2. DEFINITIONS AND APPROACH

A mobile device can capture different – often technology specific – types of network information. The most readily available information is the set of network entities currently in use for establishing connectivity. For cellular networks, these entities are called *base stations*, for 802.11 networks they are called *access points*, and for Bluetooth they are often called *devices* or *nodes*. Most wireless network interfaces also keep track of unused in-range network entities, by occasionally scanning for new ones. This information, however, is often cached and does not accurately reflect the in-range entities at all times, as the algorithms for scan initiation are optimized to reduce the scan effort: they typically only increase the scan frequency when the link to the currently used entity is weak. Furthermore, in practice this information is often not available to other parts of the device except for the network interface hardware and driver. So, for a reliable view on the in-range entities, the device must explicitly

force the network interface to execute a scan. Then, for every in-range entity, technology specific parameters are available describing such aspects as its own identity, the identity of the larger network it is part of, the strength of its link, or the operator. We are assuming here that we have an input stream that accurately describes the dynamic visibility of network entities.

We model the input stream such that it consists of dimensions with binary values (0 or 1). Every dimension represents the visibility of a network entity, where 0 indicates ‘not in range’ and 1 indicates ‘in-range’. This means that we do not consider numerical values, such as signal strengths, that are associated with network entities. As we will see, in real world circumstances, personal mobile devices have a limited amount of concurrently active dimensions – having a value of 1 – while the total number of dimensions can easily reach several thousands.

The problem definition can now be stated as follows. Let $\mathbf{A} = \{A_1, A_2, \dots, A_k\}$ be a set of attributes with A_i describing the visibility of a network entity i , and k a dynamic value growing with the number of unique network entities encountered over time. Let time be defined as a sequence $\{t_0, \dots, t_{\text{now}-1}, t_{\text{now}}, t_{\text{now}+1}, \dots\}$ with t_{now} the present time. Then

$$A_{i,p} = \{a_{i,0}, a_{i,1}, \dots, a_{i,\text{now}}\}, \quad (1)$$

is the sequence of past values, including the present value, of attribute A_i and

$$A_{i,f} = \{a_{i,\text{now}+1}, a_{i,\text{now}+2}, \dots\} \quad (2)$$

is the sequence of future values of A_i , where $a_{i,t} \in \{0, 1\}$. Consequently, \mathbf{A}_p and \mathbf{A}_f are the past and the future input streams describing the complete network visibility status from the perspective of a personal mobile device. Now, what we are interested in are changes in the status of one or more network visibility attributes: the next future event $Z_{i,f}$ where $a_{i,t-1} \neq a_{i,t}$ for $t > t_{\text{now}}$ or a set of events \mathbf{Z}_f consisting of multiple $Z_{i,f}$ s. Such a set could, for instance, be representing visibility changes of a complete 802.11 network consisting of multiple access points. In particular, we are interested in events where $a_{i,t-1} = 1$ and $a_{i,t} = 0$ because this paper deals with out-of-range prediction. Predictions are expressed as probabilities, so we want to know the probability distribution of a future event $Z_{i,f}$ given the past data: $P(Z_{i,f} | \mathbf{A}_p, H)$. Note that this distribution must be recalculated with every update of the present time t_{now} . Also, the past data encompasses only network visibility information in time, and does not include additional information at – possibly – a higher semantic level, such as the calendar information of the owner of the mobile device. H denotes the assumptions we use for determining the probability distribution, which are at this point not further specified.

When looking at human mobility, we know that nearly everyone moves according to often repeating patterns on different timescales. Many patterns, such as staying at home or at work, and traveling between home and work, take place on a daily basis. Some less frequently recurring patterns take place in a weekly fashion. We leave patterns with longer cycles than one week out of scope of this paper and therefore assume, adding to H , that visibility patterns of network entities repeat on a daily or weekly basis. The out-of-range prediction, as introduced in the next section, constructs probability density functions assuming daily cycles, where the weekly patterns may be reflected by using

different estimates for different days of the week (or sections of the week such as weekday and weekends).

3. OUT-OF-RANGE PREDICTION

The values at the present time in the input stream determine which network entities are visible. When an entity is active, we are interested in the time it becomes inactive or out-of-range. This section deals with the prediction of the end time of an active network entity, or given the begin time, the prediction of the duration.

For illustration, we use an example network entity that is visible when a person is at home. On working days, this entity is active most of the time in the evenings and all of the time during night. Figure 1 gives an overview of 15 example periods in which this entity has been active. Note that we have labeled this entity as ‘home’ for ease of understanding: our mechanism does not rely in any way on labeling entities with meaningful tags.

From this figure, it is easy to see that after midnight all intervals continue to be active until approximately 8:00 hours. The probability density function of the end of the interval, as calculated for times after midnight, therefore has a straightforward estimated form, perhaps looking like a normal distribution with a mean around 8:00 hours and a small standard deviation. But what is the end probability at 19:00 hours when an interval starts at 18:00 hours? How does this end time probability density function (pdf) evolve during the evening? We want to answer these questions in this section.

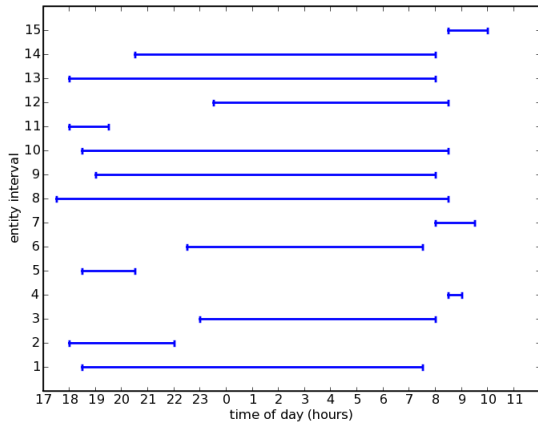


Figure 1: activation intervals I_p for an example entity

We define the current interval in which an entity is active to have a begin time t_b and an end time t_e . We are interested in the probability density of t_e to predict the *waiting time* $t_e - t_b$ for the interval to end. From this pdf it is possible to derive the probability density of the *remaining waiting time*, $t_e - t$ during the time the entity is active ($t_b < t < t_e$). Note that in general the waiting time distribution is not the same as the derived distribution, unless it has a ‘memoryless’ exponential shape. Also note that the link between these distributions is a variant on a metric from reliability theory called the hazard rate [13]. Entity intervals are denoted as I_i , with begin time of day $I_{i,b}$ and an end time of day $I_{i,e}$. The set of all previously observed intervals for an entity is denoted as I_p .

The theory of probability density estimation discerns two main approaches: *parametric* and *nonparametric*. The parametric approach assumes that the data follows a known distribution with parameters that must be chosen to best match the data. As we do not have strong hints on the model underlying the mobility of the user – a model that determines the end time probability of an entity interval – we use a nonparametric approach called *kernel density estimation*. This well known method constructs a pdf with an arbitrary shape by adding a set of *kernel functions*, one for every sample in the data set. A kernel function K is a pdf itself so that $\int_{-\infty}^{\infty} K(t)dt = 1$, and is often symmetric, with common forms being Gaussian, uniform and Epanechnikov. The type of kernel usually has little influence on the density estimates [13]; this paper uses the Gaussian kernel. The kernel method provides a continuous pdf from a fixed number of samples.

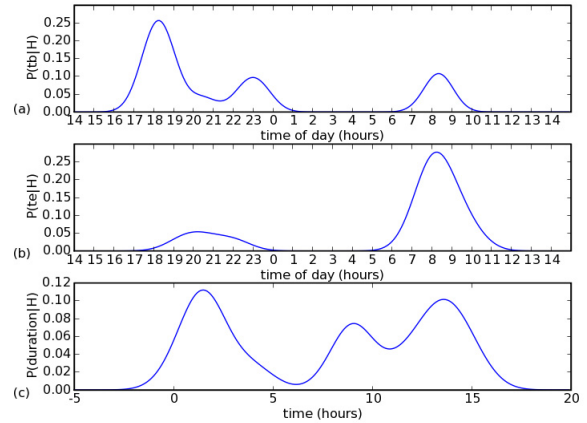


Figure 2: probability density estimates for t_b , t_e , and $t_e - t_b$, using kernel density estimation with a Gaussian kernel, and corresponding optimal h values of 0.7, 0.9 and 1.1

Now, assuming that the probability of t_e only depends on the absolute time of day and the interval duration never exceeds a single day, the estimated probability can be written as

$$P(t_e | H_1) = \frac{1}{n} \sum_{I_i \in I_p} K\left(\frac{t_e - I_{i,e}}{h}\right), \quad (3)$$

with n the number of intervals in I_p , and h the *smoothing parameter*. As in the previous section, we keep using H in this section as well to denote our assumptions. Contrary to the choice of kernel type, the value of the smoothing parameter is of high importance. A number of methods for choosing the smoothing parameter exist, but we will focus here on the *likelihood cross-validation*, for its elegance and ease of computation. This metric indicates the likelihood of the data for a density estimate based on the data set with one sample removed:

$$L(h) = \frac{1}{n} \sum_{i=1}^n \log P_{-i}(x_i), \quad (4)$$

where P_{-i} denotes the kernel density estimate with smoothing parameter h using all samples except sample i . Maximizing L gives a smoothing parameter that brings the density estimate close to the true density in terms of the Kullback-Leibler divergence.

For equation (4), $P_{-i}(x_i)$ is $P_{-i}(I_{i,e})$ using all intervals in I_p , except interval I_i . When applying equations (3) and (4) on the example intervals in Figure 1, the optimal smoothing parameter value is 0.9 and the probability of t_e is as shown in Figure 2b.

The density $P(t_e|H_1)$ predicts the current interval end time independent of the begin time. Another assumption could be that the end time does not depend on the absolute time of day but rather on the – relative – duration of the interval. In that case, the end time probability can be calculated using the duration probability and the known time of day value t_b . Using the same method as for $P(t_e|H_1)$, $P(t_e - t_b | H_2)$ or $P(duration|H_2)$ has an optimal cross-validated likelihood when h is 1.1, for the example data of Figure 1. This density is depicted in Figure 2c. Note that a negative duration does not always have a zero probability, which shows a known disadvantage of the kernel density method. For bounded domains, this may occur when many samples have values close to the boundary. Although existing approaches tackle this problem successfully, they come at a cost of, for instance, a more elaborate kernel [5]. In the case of a bounded variable with the majority of samples far from the boundary, this effect has no importance. For now, we will ignore this effect.

Let us estimate $P(t_e | t_b, H_3)$, the probability of t_e assuming that it depends on t_b . This seems a fair assumption, when looking at the data in Figure 1. However, now we can not use a straightforward kernel density estimate anymore, but have to switch to *kernel conditional density estimation* [3]. This method allows us to generate a continuous pdf using a double kernel estimator, weighting both variables with their own smoothing parameter. The interval end time probability given the begin time is now

$$P(t_e | t_b, H_3) = \frac{\sum_{I_i \in I_p} K\left(\frac{t_e - I_{i,e}}{h_1}\right) K\left(\frac{t_b - I_{i,b}}{h_2}\right)}{\sum_{I_i \in I_p} K\left(\frac{t_b - I_{i,b}}{h_2}\right)}, \quad (5)$$

with h_1 the smoothing parameter for the end time and h_2 the smoothing parameter for the begin time. For this kernel estimate we can again turn to the leave-one-out cross validation to select the optimal h_1 and h_2 [3]. It is based on the probability $P(I_{i,e} | I_{i,b} | H)$ of an interval I_i occurring given the parameters h_1 and h_2 . The cross-validated log likelihood for the kernel conditional density estimate is defined as:

$$L(h_1, h_2) = \frac{1}{n} \sum_{i=1}^n \log(P_{-i}(I_{i,e} | I_{i,b}) P_{-i}(I_{i,b})) \quad (6)$$

When using this equation on the example data of Figure 1, the optimal parameter values are $h_1 = 0.8$ and $h_2 = 0.7$. The conditional probability density is depicted in Figure 3 in a 3 dimensional form and a contour form. It clearly shows that the shape of the end time event density varies considerably given different begin times and therefore offers more information than can be extracted from the accumulated density in Figure 2b.

We now have three models for predicting the interval end time and can compute which model fits best with the data. The log likelihood of the end times given the corresponding begin times and model is $\sum_i \log P(I_i | H)$. For $P(t_e|H_1)$ its value is -27.1, for $P(duration|H_2)$ it is -37.5 and for $P(t_e | t_b, H_3)$ it is -19.8, which makes the prediction of t_e assuming that it depends on t_b the best

model for the example dataset. Other data sets may result in a different preferred model, for instance when a small set of intervals occur randomly during the day and have very similar durations, in which case the relative duration model fits best.

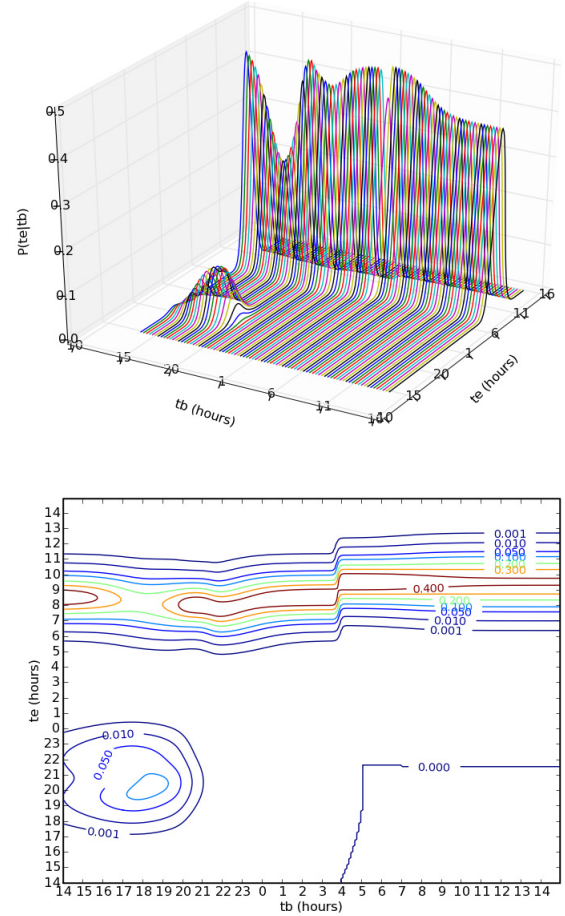


Figure 3: 3d plot and contour plot of the conditional probability $P(t_e|t_b, H_3)$ using a kernel conditional density estimation with a Gaussian kernel and optimal parameters $h_1 = 0.8$ and $h_2 = 0.7$

In the calculations above, we do not take into account the weekday, so that the weekly patterns are not represented. We can compute, however, different densities for different days of the week, or make a distinction between weekdays and weekend days.

Now, the remaining waiting time can simply be derived by taking the waiting time density divided by the integral over the waiting time density from t to infinity – scaled to always integrate to unity. For the conditional density estimate in (5), this becomes

$$P(t_e | t_b, t, H_3) = \frac{P(t_e | t_b, H_3)}{\int_t^{\infty} P(t_e | t_b, H_3) dt_e} \quad (7)$$

Obviously, limits may behave badly (when the integral is going to zero).

In summary, we now have a set of tools to estimate the probability density of out-of-range events, consisting of kernel

density estimation, kernel conditional density estimation, as well as a metric to find the optimal smoothing parameters. These tools allow us to compare and select different models.

4. USER EXPERIMENT

We have collected traces during a user experiment with 12 participants, from February 1st until March 14th 2007. The main objective was to gather accurate visibility information on a personal mobile device for as many as possible network technologies. Although existing traces on mobile devices exist in the public domain (see [1], [12], [10]), we are not aware of data sets that contain information on more than two technologies: our traces capture cellular, 802.11 and Bluetooth events. The participants were all Telematica Instituut colleagues, some working mostly at the office and others more often traveling.

The platform for collecting the multi-network traces is based on the Network Abstraction Layer (NAL) software described in [11]. The NAL implements a cross-layer network resource abstraction, connecting different elements such as network interfaces, links, paths, gateways, etc. in a formal manner. It communicates updates on the state of the network resources – here on link and network layer – to subscribers by means of messages. Messages arrive in a light-weight transaction context to ensure state consistency.

We want to capture network traces on personal mobile devices that are carried by the user during their daily activities. We selected Windows CE as operating system and chose two devices to use during the experiment: the HTC Wizard (Qtek 9100, T-Mobile MDA, etc.) and the HTC Prophet (Qtek S200, i-mate JAMin, etc.), both running Windows Mobile 5.0. These devices have network interfaces for cellular GPRS, 802.11 wireless LAN, Bluetooth, fixed serial USB, and infrared communication. We used existing NAL modules for cellular, 802.11 and Bluetooth monitoring, as well as IP configuration notifications.

The cellular module reflects changes in the currently used cell by means of notifications. It cannot provide information about in-range nearby cells, but can indicate operator names of available cellular networks. Operator names are scanned every 5 minutes; cell id notifications are asynchronous – are generated at the moment of change. To run a successful experiment, the NAL needed several extensions. For 802.11 logging, the 802.11 network interface must be occasionally activated after which the Windows CE Wireless Zero Configuration (WZC) starts scanning for available networks and initiates 802.11 association according to the default wireless LAN preferences of the user. Once every 10 minutes, the 802.11 interface is active for 1 minute. Within that minute, the NAL generates events reflecting the scanning results and possible associations. The NAL Bluetooth module takes care of scanning to discover in-range nodes. This scan takes place every 5 minutes. We do not scan continuously because that consumes too much energy and interferes too much with other user initiated Bluetooth usage such as GPS device connections.

One hard part of the measurement platform implementation was the handling of the power state of the 802.11 interface. To preserve power, the wireless LAN interface cannot be always on: this would deplete the battery in a few hours. By applying a 1 to 9 on-state ratio, the participants on average needed to recharge their device once a day. Although still much more often than under normal operating conditions, this is a workable situation in which participants can charge their devices at night. Another difficult

part of the platform implementation was the management of the overall power state transitions. When the user pushes the power button to shutdown the display, the device goes into suspend mode. In this state, the CPU does not execute instructions and the operating system only resumes after a hardware interrupt. In such a state, the measurement platform cannot process NAL notifications or initiate the activation of a network interface in a regular interval. This means that the platform needs to make sure that the device will not go into suspend mode. To further complicate the power management, the 802.11 interface can only be activated when the device power state is fully on. Also, the cell id updates are only generated reliably if the power state stays above a certain value. Our software tackles all these issues successfully.

The measurement software was installed on the participant’s devices in a matter of a few days. After the software installation on the device, the participant immediately started with the experiment, which means that they did not start and stop at exactly the same time. In week 1 and week 5, the participants received questions investigating the interruptability of the participant [4]. After roughly five weeks, we collected the data from the SD cards and uninstalled the measurement platform¹.

During the execution of the experiment, a number of problems occurred. One participant stopped after 14 days due to software stability problems. This participant used a device with a different operating system version. Another three participants did not log network resource information during the first two weeks of the experiment due to a bug triggered on non-English language devices. Participant 8 experienced several clock resets to the extent that it is not possible to restore the chronological order of the logged events. Most participants experienced a device crash every few days, which interrupted the logging for up to a few minutes.

For 11 participants, not counting participant 8, the accumulated experiment duration is 373 days with an average duration of 33 days per participant. The basic numbers for the amount of global network entities is shown in table 1.

Table 1: basic global network entity count

unique used cells	4120
different in-range operators	18
unique in-range 802.11 access points	3787
unique in-range 802.11 networks	1725
unique in-range bluetooth nodes	6679

The number of unique cells and the number of different in-range operators looks large at first impression. But, the participants cover all five active operators in the Netherlands, Telematica Instituut is close to the German border, and some participants have been abroad in Germany, Belgium and the UK during the trial period. Also, the variation between participants is quite large, indicating a big difference in traveling patterns.

The individual number of unique in range 802.11 access points also shows a large variation between participants, although a

¹ Software and data overview available at <http://cosphere.telin.nl/>

correlation does not look strong: for instance, participants 4 and 9 have a below average number of access points and an above average number of cells.

The definition of an 802.11 network is based on the name of 802.11 SSID: all access points with the same SSID are counted as being in one network. In practice, these access points may not actually form one network, but may coincidentally have the same name.

5. PREDICTION EXAMPLE

In this section we apply – by means of example – the density estimation method introduced in section 3 to the data collected during the user experiment. Suppose we have a data synchronization job that needs to use a network resource for an uninterrupted long duration, say one hour. Then, we would prefer to only use those network entities that have a low probability of losing visibility in the next one hour. So, as example, we now want to determine all the occasions during weekdays that for a visible network entity the probability of getting out-of-range in the next one hour is lower than, say, 0.1. We assume that the type of the network entity is irrelevant, so that cellular, 802.11 and Bluetooth entities are all considered. By doing so, we are capable of comparing and characterizing the different technologies. Also, for sake of simplicity, we do not take into account that some entities form larger networks that already offer uninterrupted connectivity below the network layer, such as office wireless LANs. Indeed, if we did, this would create more occasions with low probability in interruption.

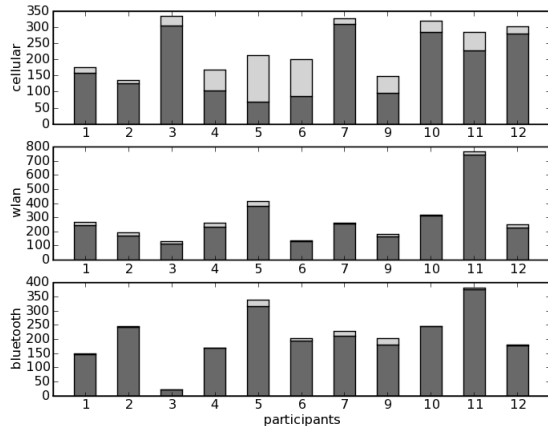


Figure 4: number of matching intervals (light grey) and non-matching intervals (dark grey) for cellular, 802.11 and Bluetooth entities during the trial (per participant)

As starting point, we take the conditional pdf in equation (7), which means that we must extract the begin- and end-time for all the weekday intervals for all the entities from the input stream. We have made a minor adjustment on the input stream by removing very short intervals (< 10 minutes) and concatenating intervals with a very short interruption (< 5 minutes). Consequently, we start computing the probability of losing visibility for the next hour only after 10 minutes of interval visibility. The calculation of the probability density function using kernel (conditional) density estimates consists of two main steps. First, the optimal parameters h_1 and h_2 must be determined

for every entity. As this is a time consuming process, we only use the top 5 most-visible entities for every technology, in terms of total accumulated duration (not in terms of number of visibility intervals). The algorithm for likelihood cross-validation as given by equation (6) is straightforward: we used a range of [0.1, 10.0] to determine the best smoothing parameters, with values typically around 1.0. At step 2 we compute the actual probability by integrating over (7) one hour forwards. The bottom term in (7) goes to infinity, so we need to find a suitable cutoff point. We have set this to the maximum end time in the set plus $4 * h_1$, as $e^{-(4h_1/h_1)^2/2}$ is sufficiently small (remember: we are using a Gaussian kernel). The integration delta depends on the level of smoothing and is set to $h_1 / 10$. The probability is calculated for every interval for every top 5 entity, in steps of 15 minutes, starting after the first 10 minutes of visibility: if the current time gives a probability larger than 0.1, we increment the time with 15 minutes and recalculate. If the probability is below 0.1 for any time, the interval is marked as a match. Note that the set of intervals also contains the current interval, and the optimal smoothing parameters are calculated using the full set. This has little impact, however, because the sets of intervals typically consist of 30 or more samples. When doing this for all top 5 entities, we get a result as depicted in Figure 4, where the light grey parts of the bar are intervals that have sometime during their existence a probability of less than 0.1 to get out-of-range in the next hour. The dark grey parts have not. It clearly shows that cellular, and to a lesser extent, 802.11 entities have more of matching intervals. Translating this to the number of matches per day, we see the result as depicted in Figure 5.

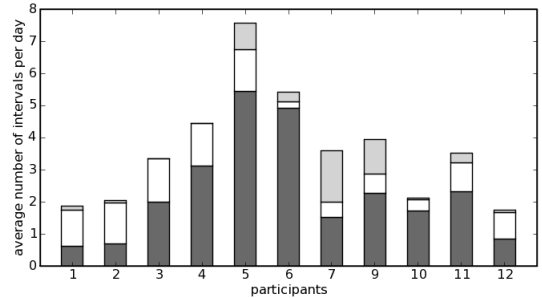


Figure 5: average number of sync opportunities per day for cellular (dark grey), 802.11 (white) and Bluetooth (light grey) per participant

As mentioned in section 3, bounded domains are sometimes less well estimated with kernel densities. We have not compensated here by using an alternative kernel method. However, this disadvantage only occurs with many samples close to the boundaries, for instance when intervals for an entity are always short and end times are close to the begin times. In these cases, it is likely that the probability density is already high (but not as high as it should be) and therefore will not be a match in our example.

6. RELATED WORK

A few recent papers have a full or partial focus on the timing aspects in the prediction of network visibility, while a number of other papers are related in the sense that they predict network visibility from another point of view.

The work presented in [8] explicitly focuses on the transient behavior of access point association by using a semi-Markov model. The density estimation is based on a histogram approach which is known to be less accurate than kernel density estimation for many situations. Furthermore, the out-of-range prediction is based on the interval *duration* of previous access point associations, which is also the case in [9], and does not incorporate a conditional parameter as we do or vary the density estimate with the time of day. This may influence the density estimation accuracy. The approach described in [9] is partially based on parametric density estimation where residence time is fitted to a Weibull distribution. Both papers use the same data set (Dartmouth) collected at the infrastructure side.

The authors in [14] compare a number of prediction algorithms using real-user 802.11 infrastructure traces. The prediction concerns the next access point given a history with a fixed or dynamic length, and includes an n-order Markov model approach. Our approach differs in the sense that our primary interest is in the time aspects of visibility and not the logical order. Also, our dataset is produced on the device side, not in the infrastructure, and therefore gives a more complete view on what is visible from the mobile device perspective, although it is by far not as large as their 802.11 traces. The method presented in [7] focuses on predicting routes between ‘bases’ using GSM cell traces. Paths between routes are clustered together when they are similar. Time aspects such as the duration of routes and length of stay in bases are not incorporated.

Although strictly speaking the work in [2] does not cover wireless networks, it is concerned with recognizing ‘trajectory patterns’ and explicitly models the time it takes to go from one location to another. It does not, however, take into account the time spent at a location, let alone the probability density of the duration.

7. CONCLUSIONS

We have introduced a method for predicting out-of-range events of network entities on a personal mobile device, based on kernel density estimation. We have gathered real-user mobility traces using three different network technologies on a mobile device, and have used these traces to illustrate the density estimation possibilities with an example.

Obviously, the dynamics in the network resources as observed during this experiment depend on many aspects: the selected group of participants, the society and environment in which the participants live, their profession, the period in time, etc. We do not aim here on disclosing universal mobility characteristics, but rather provide a mechanism for modeling the network surroundings of individual users.

Although the focus in this paper has been rather narrow, it is clear that using conditional probability density estimates can be applied in other ways. For instance, this method could also contribute to the prediction of the begin time of visibility intervals, using another conditional such as one or more previously visible network entities.

We plan to release a public version of the data in anonymized form at a later stage.

8. ACKNOWLEDGMENTS

We would like to thank Henri ter Hofte and Raymond Otte for their contributions to the definition and the construction of the platform software, as well as for their support during the execution of the experiment. Also, we would like to thank Rogier

Brussee for his feedback on the density estimation approach. The Dutch Freeband Communication Research Programme (Awareness project) supported this research under contract BSIK 03025.

9. REFERENCES

- [1] N. Eagle, and A. Pentland, Reality mining: sensing complex social systems, *Personal and Ubiquitous Computing*, 10, 4 (March 2006), 255-268
- [2] F. Giannotti, M. Nanni, D. Pedreschi, and F. Pinelli, Trajectory Pattern Mining, In *Proceedings of the Conference on Knowledge Discovery and Data Mining (KDD'07)*, San Jose, USA, August 2007
- [3] M. Holmes, A. Gray, and C. Isbell, Fast Nonparametric Conditional Density Estimation, In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI'07)*, 2007
- [4] H. ter Hofte, Xensible Interruptions from Your Mobile Phone, In *Proceeding of the International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI'07)*, Singapore, Sept. 2007
- [5] M. Jones, Simple Boundary Correction for Kernel Density Estimation, *Statistical Computing*, Volume 3, Number 3, pp. 135-146, 1993
- [6] M. Kijima, Markov Processes for Stochastic Modeling, Chapman and Hall, 1997
- [7] K. Laasonen, Clustering and Prediction of Mobile User Routes from Cellular Data, In *Proceedings of the Conference on Principles of Data Mining and Knowledge Discovery (PKDD'05)*, October 2005
- [8] J-K Lee, and J Hou, Modeling Steady-state and Transient Behaviors of User Mobility: Formulation, Analysis, and Application, In *Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'06)*, May 2006
- [9] D. Lelescu, U. Kozat, R. Jain, and M. Balakrishnan, Model T++: An Empirical Joint Space-Time Registration Model, In *Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'06)*, May 2006
- [10] M. McNett, and G. Voelker, Access and Mobility of Wireless PDA Users, *Mobile Computing and Communication Review (MC2R)*, 9, 2 (April 2005), 40-55
- [11] A. Peddemors, H. Eertink, I. Niemegeers, and M. Bargh, Network Resource Awareness and Control in Mobile Applications, *IEEE Internet Computing – Special Issue on Roaming*, 11, 2 (March/April 2007), 34-43
- [12] A. Rahmati and L. Zhong, Context-for-Wireless: Context-Sensitive Energy-Efficient Wireless Data Transfer, In *Proceeding of the Conference on Mobile Systems, Applications, and Services (MobiSys'07)*, June 2007
- [13] B. Silverman, Density Estimation for Statistics and Data Analysis, *Monographs on statistics and applied probability*, Chapman and Hall, 1986
- [14] L. Song, D. Kotz, R. Jain and X. He, Evaluating location predictors with extensive Wi-Fi mobility data, In *Proceedings of INFOCOM*, April 2004