A Model for the Online Time of Network Users

Matthias Heymann
Bell Laboratories, Statistics Department,
Murray Hill, New Jersey, USA
mail@MatthiasHeymann.de

Mark H. Hansen
Bell Laboratories, Statistics Department,
Murray Hill, New Jersey, USA
cocteau@research.bell-labs.com

August 29, 2003

Abstract

In this paper we analyze network traffic. While in most existing papers data from many users has been seen from the server's point of view, we analyze data of individual users.

Based on our observations that we made with our visualization tool we give a reasonable definition of the time a user spends online. We present a model for these online times and a fitting routine that performs well for users with different kinds of working schedules. This approach might help understand the structure of traffic on the server side.

1 Introduction

Since the Internet has become more and more important in our daily lives, statisticians and industry are interested in understanding the structure behind everything that happens between server and surfer.

On the server side it is important to know how to deal with the huge amounts of data that are sent to the client computers with as little delay as possible. When new servers are constructed, they can be tested with programs such as SURGE [1] that simulate traffic of a given number of users with certain profiles. In order to be flexible when performing such tests, it is not enough just to replay a previously recorded traffic logfile. Instead, much work is done to model the traffic on the server side and to find its most significant parameters. These models are often inspired by the idea that the data is the superposition of data from many independent sources.

We want to go one step back and get an insight of the behavior of only one individual user. This might help for future work on modelling the mixture of the data.

2 The Data

Our data was collected between 11/14/2001 and 01/17/2002 and consists of 2 million connections to the Bell Labs server. Since we decided to focus on the description of the start and end times of the users' computer sessions, we used only the following information for each connection: The ip-address of the user, the start and end times of the connection and the port number used for the connection. While in our data the ip-addresses correspond one-to-one to the users, the port numbers describe the protocol used for the connection which in turn corresponds to the application that invoked the connection. For further explanations on the basic expressions concerning internet traffic we refer the reader to the introductory text [2].

Our original data set contained 6000 different ip-addresses. In a first step we decided to keep only those of the form 135.104.50.xxx because this guarantees that the data comes from Bell Labs employees working from their homes, and not just from computers running some automated processes. In a second step we kept only those ip-addresses with more than 10 kB of data because less data did not seem enough for us to visually identify repeating patterns. Using the visualization tool described later in this article, we then looked at the remaining 70 users and decided to keep only 20 of them who provided us with data from sufficiently many days, and that showed behavior that was regular enough to run our fitting algorithms on them.

Of course the number of remaining users is too small to make general statements on user behavior, but it is enough to see that our model works well for a variety of users who show some regularity in their working schedules. More data would be necessary to continue this work, for example by clustering users based on the parameters of our fitting process, or by building a hierarchical model that also includes the usage of certain ports. For this one would have to keep in mind that our data comes from a non-representative group of people who use their computers in a much more sophisticated way than the common internet surfer does.

3 Definition of Online Time

Given the data described above, it is not at all clear when a user is online and when he is not. It is possible that he surfs on a website that is cached on his computer so that no new connections are established. Also, he might be busy reading a website for a while and therefore not request data from the server. Consequently, we need a reasonable definition of online time, given the data of one individual user. We decided to use the following procedure:

First we defined the online time for each single port in four steps, later we would merge all ports together. In step 1 we merged all overlapping connections. In step 2 we filled all pauses of less than a second, which also led to a drastic reduction of the memory required for our data. The online times defined so far are plotted as colored blocks in our visualization tool (see section **??** and Figure

XXX).

In step 3 we took care of the typical appearance of periodic behavior on certain ports. For example, users oftentimes activate a function in their email program to check for new incoming emails in certain time intervals, and we will see this as short regularly appearing connections on the pop3 port. During the time inbetween such connections the user has not closed his email program, otherwise the pattern would be interrupted. Therefore we coded an algorithm that detected such patterns and defined the time inbetween also as online time, as long as the connections are less than 2 hours apart. We chose the 2 hour bound because periodic behavior in the online times with low frequencies can be captured by our model (see our last example).

In step 4 we deleted again pauses shorter than a certain time ΔT, which captures the idea mentioned at the beginning that for short pauses it is not likely that the user really closed his program. For our data set $\Delta T = 17$ minutes seemed to be a good compromise that does not oversimplify the data.

Finally, we merged the online times of all the single ports and took this as the final definition of the overall online time.

The online time derived from this procedure is drawn in light grey in the background of our visualization tool.

4 Our Visualization Tool

To get a good idea about the nature of our data, and to find possible problems to solve, we started our work by writing a program to visualize the data of individual users. First we used Pearl for all routines that were either very time-consuming or that had to deal with very large data sets. Once all preprocessing was done, we used R for our interactive visualization tool, and later for our fitting algorithms and our simulation routines.

Figure XXX shows a screenshot of our visualization tool displaying the data of one user. Every horizontal line represents the online times of the port given by its port number on the left and the port name on the right of the line. The upmost line represents the mixture of all those ports that are not displayed in the other separate lines, and the last line is the mixture of all ports. The tool runs in two different modes: In mode one the single ports that are displayed in separate lines are the ones that are invoked most in the data of *all* users, while in mode two those ports are shown that are used most by the observed user only. The mode can be switched by clicking on the mode button.

In every line the colored blocks represent times (as calculated by the steps 1 and 2) in which connections on the corresponding port(s) are open. Since the border of the blocks are darker the interior, a dark block appears whenever there is a sequence of very short connections that cannot be separated on the screen. The online times as calculated by the steps 1 - 4 appear in light grey in the background.

The tool allows us to see the data at various time scales from several weeks to just a couple of seconds. Initially all data is displayed. Then we can zoom

in by clicking twice in the plotting area, choosing the left and the right border of the next plot. The arrow buttons on the upper right of the plot let us browse through the history of our series of graphs for this user, just as in every common webbrowser. The measure button allows us to accurately measure time differences between certain events by clicking on the button and then twice into the plotting region. The other three buttons on the left draw plots that we were interested in during our investigations, such as distributions of the pause lengths, the interarrival times and the session durations.

To give an example, let us now take a closer look at Figure XXX: We see a user's computer session from about 11:30 am to 1 am. He starts by opening his webbrowser and uses the http port 80. At the same time (and probably caused by this), two overlapping sequences of connections on port 53 (the domain name server) start. Our frequency analysis algorithm could capture them both, and therefore it defined the time of those sequences as online time on this port, as drawn light grey in the background. The periodic behavior in port 445 could not be detected from the beginning, since measurements show that the time between the first two connections is considerably longer than the following time differences.

Just minutes later the user opens his email program which can be seen from the IMAP connection (port 143), and writes some emails (port 25). After some time of less activity between 4 pm and 7 pm he starts using ftp. At 1 am he writes a last email and closes all his programs within 20 seconds (the exact order can be determined by zooming into the plot).

5 Graphical Representation of Sessions

Now after defining the online time, we needed to find a good graphical representation for it in order to test if our models are performing well, that means if the data from a simulation looks similar to the original one. We found two characteristic representations, as seen for example in Picture 1:

In the first type of graph every row represents one day, and we marked the time spent online. This is the most intuitive one, and one can immediately get an insight into the user's typical working schedule. In the second type of graph every session is represented by a point whose x-value is given by the start time and whose y-value is given by the end time of the session. With this one can get an idea of how often certain start-end-combinations appear in the data.

Picture 1 is an example of the two graphs of a user who shows a very regular behavior: He works from about 9 am to about midnight without any interruption. This can be seen in both graphs. The user in Picture 2 works in two sessions: The first one goes from about 9 am to 6 pm and the second one from about 6 pm to midnight, with a short dinner break inbetween. One can also see that both users take a week of holidays, where the holidays of the second one are exactly between Christmas and newyear.

After looking at several users we found that the second type of graph shows an aspect of the data that we took into account later when we constructed our

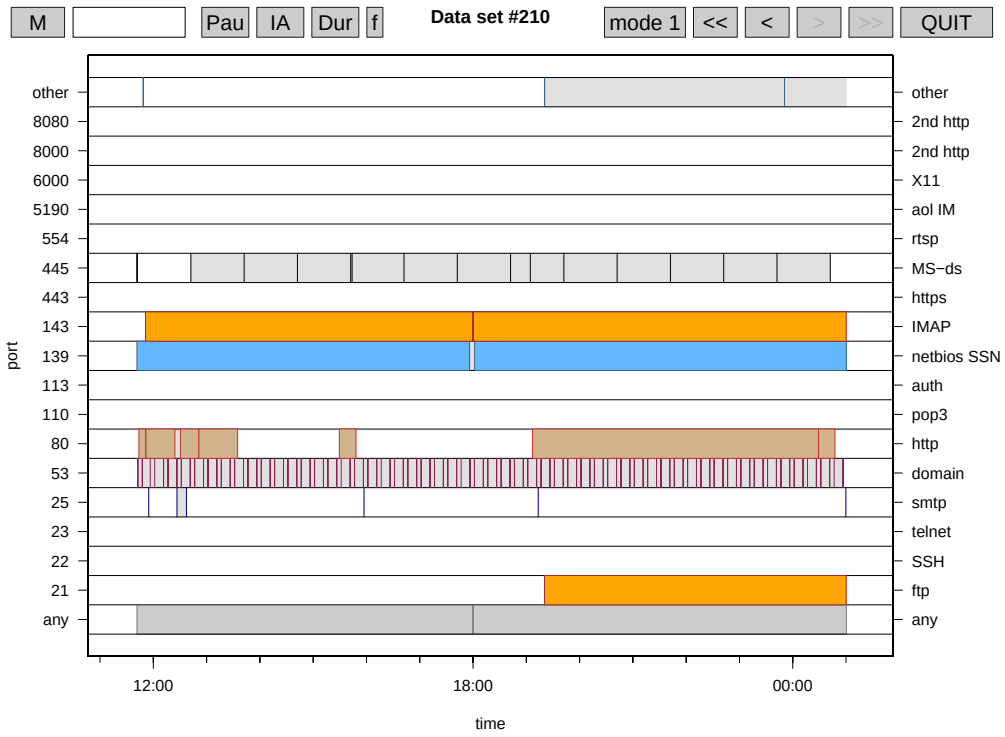

Figure 1: A screenshot of our visualization tool.

model: There are two different classes of sessions. In sessions of type A the user just wants to finish a task and then end the session after a short time. These sessions can be found on the diagonal since start and end time are almost the same. In sessions of type B he follows his personal working schedule and ends his session at one of the end times that are characteristic for this user, depending on the start time of the session. Sessions of this type form horizontal patterns in the graph.

6 A Model for Online Times

Let a_i and b_i be the start and end times of session i, respectively ($i = 1, \ldots, n$). For simplicity we assume that there are no connections open at the beginning of the data collection, and that the last session is finished before its end at time T.

We also assume that the distribution $f_E(\cdot|a_i)$ of the end time of a session depends only on its start time, and that the distribution $f_S(\cdot|b_{i-1})$ of the start time of a session depends only on the end time of the previous session.

Therefore we can write

$$\begin{aligned} f(data) &= \prod_{i=1}^{n} f_S(a_i|b_{i-1}) f_E(b_i|a_i) \cdot p, \\ p &= \exp(-\int_{b_n}^{T} \Phi_{\text{on}}(t)dt). \end{aligned}$$

where $f_S(a_1|b_0)$ is the probablility that a_1 is the first start time after the beginning of the data collection and p is the probability that there is no start of a session between b_n and T.

When modelling f_S we were inspired by the idea of an inhomogeneous point process that starts new sessions. So given an end time b_{i-1}, we would run a point process with a time-dependent, 24-hour-peridic rate function Φ_{on} and pick the first event as the start a_i of the next session. Following this construction, f_S is given by

$$f_S(a_i|b_{i-1}) = \Phi_{\text{on}}(a_i) \exp(-\int_{b_{i-1}}^{a_i} \Phi_{\text{on}}(t)dt).$$

Note that for a constant rate function $a_i - b_{i-1}$ is exponentially distributed, and if Φ_{on} is piecewise constant then the probability density of $a_i - b_{i-1}$ is piecewise exponential.

f_E is a mixture of two models, one for sessions of type A and one for sessions of type B. Given the start time a_i of a session, the probability to have a session of type A is $\pi(a_i)$, the probability for a session of type B is $1 - \pi(a_i)$. The time dependency of these probabilities is due to the fact that sessions of one type might be more likely at different times of the day. For example, a user might have the habit to check his email at some point after work and then end his session again, so that type-A-sessions are more likely in the evenings.

Since quantile plots for the durations of the sessions showed that the short durations are approximately exponentially distributed, we modelled the durations of type-A-sessions exponentially with a time-independent parameter λ. For type-B-sessions we used the same construction as for the start times, with a rate function Φoff.

Denoting the overall offline time by off, we derive the density function of a dataset D:

we now find that the density function for a dataset is given by

$$\begin{aligned} f(D) &= \prod_{i=1}^{n} \Phi_{\text{on}}(a_i) \cdot \exp\Big(- \int_{\text{off}} \Phi_{\text{on}}(t)dt \Big) \\ &\quad \cdot \prod_{i=1}^{n} \Big[\pi(a_i)\, e^{\lambda(b_i - a_i)} + \big(1 - \pi(a_i)\big)\, \Phi_{\text{off}}(b_i) \exp\Big(- \int_{a_i}^{b_i} \Phi_{\text{off}}(t)dt \Big) \Big] \end{aligned}$$

7 The Fitting Process

Now given the data of a user, we need to fit the parameters of the model, that is $\Phi_{\text{on}}(t)$, $\Phi_{\text{off}}(t)$, $\pi(t)$, and λ. For simplicity we choose all the time-dependent parameters piecewise constant (on the same intervals for $\Phi_{\text{on}}(t)$ and $\pi(t)$):

$$\Phi_{\text{off}}(t) = \sum_{j=1}^{M} \beta_j 1_{I_j}(t), \qquad \Phi_{\text{on}}(t) = \sum_{k=1}^{N} \gamma_k 1_{J_k}(t), \qquad \pi(t) = \sum_{k=1}^{N} \pi_k 1_{J_k}(t),$$

with 24-hour-periodic sets I_j and J_k that being determined during the fitting process. To maximize the log-likelihood-function, we find an explicit formula for the estimators of the γ_k:

$$\hat{\gamma_k} := \frac{\sum_{i=1}^{n} 1_{J_k}(a_i)}{|J_k \cap \text{off}\,|}.$$

Since the second part of the density function consists of a mixture model, we use the EM algorithm to estimate the remaining parameters. Starting from arbitrary parameters π, λ and β, we estimate the probability that session i is of type X by

$$\tau_{X,i} := \frac{\pi^{(X)}(a_i) f^{(X)}(b_i|a_i)}{\sum\limits_{X \in \{A,B\}} \pi(a_i)^{(X)} f^{(X)}(b_i|a_i)} \qquad (X = A, B, \quad i = 1, \ldots, n).$$

In the M-step we also find explicit formulas for the next estimators:

$$\begin{aligned} \hat{\pi}_k^{(X)} &:= \frac{\sum_{i=1}^{n} 1_{J_k}(a_i)\tau_{X,i}}{\sum_{i=1}^{n} 1_{J_k}(a_i)}, \\ \hat{\beta}_j &:= \frac{\sum_{i=1}^{n} 1_{I_j}(b_i)\tau_{B,i}}{\sum_{i=1}^{n} |I_j \cap [a_i, b_i)| \cdot \tau_{B,i}}, \\ \hat{\lambda} &:= \frac{\sum_{i=1} \tau_{A,i}}{\sum_{i=1}^{n} \tau_{A,i}(b_i - a_i)}. \end{aligned}$$

For every fixed set of intervals I_j and J_k we started the EM algorithm several times and picked the parameters with the largest likelihood. Starting from these intervals we then tried all possibilities of splitting any of these intervals into two new ones and looked for a better set of parameters. Here we restricted our choices of intervals to those whose length is a multiple of 30 minutes. After we reached a total number of 30 intervals, we started to merge neighbored intervals subsequently and again to look for better parameters. At the end we compared all our solutions for the various choices of intervals. To avoid overfitting, we used the BIC to determine the best number of intervals.

You can see one possible outcome of this procedure in picture X where we plotted the best BIC over the number of intervals. Starting from only two intervals, it decreases until the improvement in the maximum likelihood weighs less than the term added by the BIC strategy so that the value increases again. The second branch of the graph is the result of the second part of the algorithm which usually improves the total result again.

8 The results of the fitting process

After applying this algorithm to several users we were surprised how flexibly it recognized different kinds of regularities in the data.

Picture YYY shows the graphs of a user with a fairly easy schedule, starting at about 8 am and finishing shortly before midnight. The parameter $\lambda(t)$ seems overfitted, the reason for this being revealed by the graphs for the simulated data: The algorithm regarded only short sessions as sessions of type B (which was also our idea when we introduced the two types of sessions), and there are only very few such sessions in this dataset.

The graphs for the simulated data look very much like the original one. The only difference is that the free days in the simulated data are spread evenly over all the seven weeks. Including the idea of holidays into the model should not be difficult.

The fitting results for the second test user was similarly satisfying: The algorithm recognized the fact that the user makes dinner breaks at about 6 pm. Another interesting aspect is the number of times that the user works long at night, which in this case is similar in the original and in the simulated data.

We were then curious how well the algorithm would perform on less regular data. The user in Picture ZZZ does not have a regular working schedule except for the fact that he rarely works at night. It turned out that the fitting algorithm not only captured this fact, it also used sessions of type B to simulate the short ftp connections that frequently appear at 3 am.

9 Conclusions and Future Work

We programmed a tool for visualizing data of network users. We then proposed a way to define a user's online time, built a model for it and used the data to

fit its parameters. The outcome of a simulation routine shows that the model performs well in various situations in which the data used for the fitting has some kind of regularity.

With more data available, the results of the fitting process could be used for clustering techniques that identify groups of similar users. Such groups could be characterized for example by the number or times of the working sessions, or by the average online time per day.

Another possible way to procede would be to use our results as the highest level of a hierarchical model that in lower levels also includes the various ports and then possibly even single connections. After that, the superposition of the simulated data from single users can be compared to the various other ways of simulating data on the server side.

References

[1] P. Barford and M.E. Crovella, *Generating Representative Web Workloads for Network and Server Performance Evaluation*, in Proceedings of Performance '98/ACM SIGMETRICS '98, pp. 151-160

[2] William S. Cleveland and Don X. Sun, *Internet Traffic Data*, Journal of the American Statistical Association, 95, 979-985, 2000. Reprinted in Statistics in the 21st Century, edited by A. E. Raftery, M. A. Tanner, and M. T. Wells, Chapman & Hall/CRC, New York, 2002

[3] Internet Security Systems, `http://www.iss.net/security_center /advice/Exploits/Ports/default.htm`

[4] The Internet Spec List, `http://www.graphcomp.com/info/specs/ ports.html`

[5] D.R. Cox and Valerie Isham, *Point Processes–Monographs on Applied Probability and Statistics*, Chapman and Hall, 1980

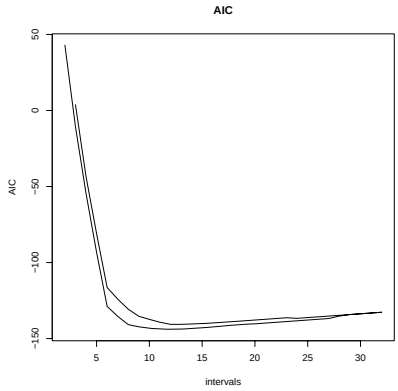

Figure 2: This graph shows how the right number of intervals is determined. The AIC criterion prevents from overfitting by using too many intervals.

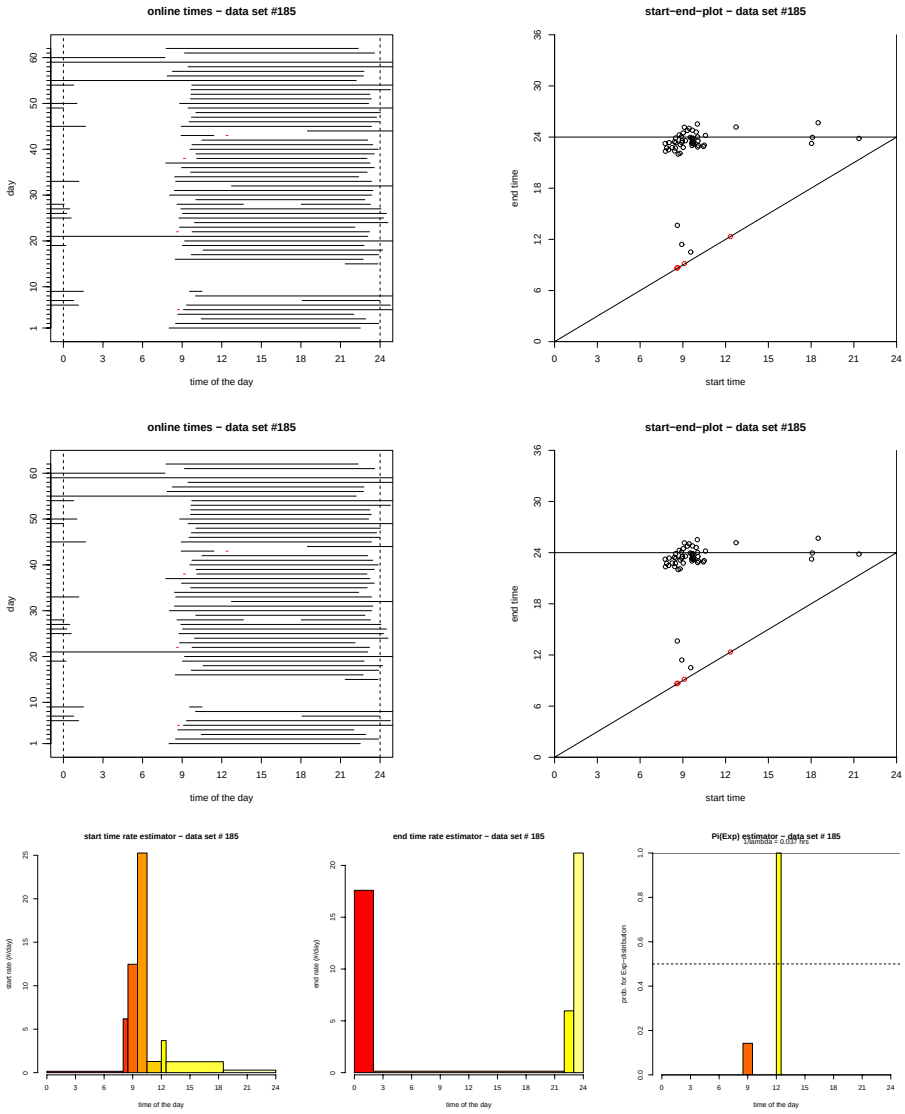

Figure 3: First row: The two plots for the original data of a test user. Red color indicates that the fitting m determined them as sessions of type B with probability > 0.5.
Second row: (XXX still original data XXX) Data from a simulation using the parameters derived from the fitting process.
Third row: The results of the fitting process: The rate for the start time has a peak at 8 am, the rate for the end times at about midnight.

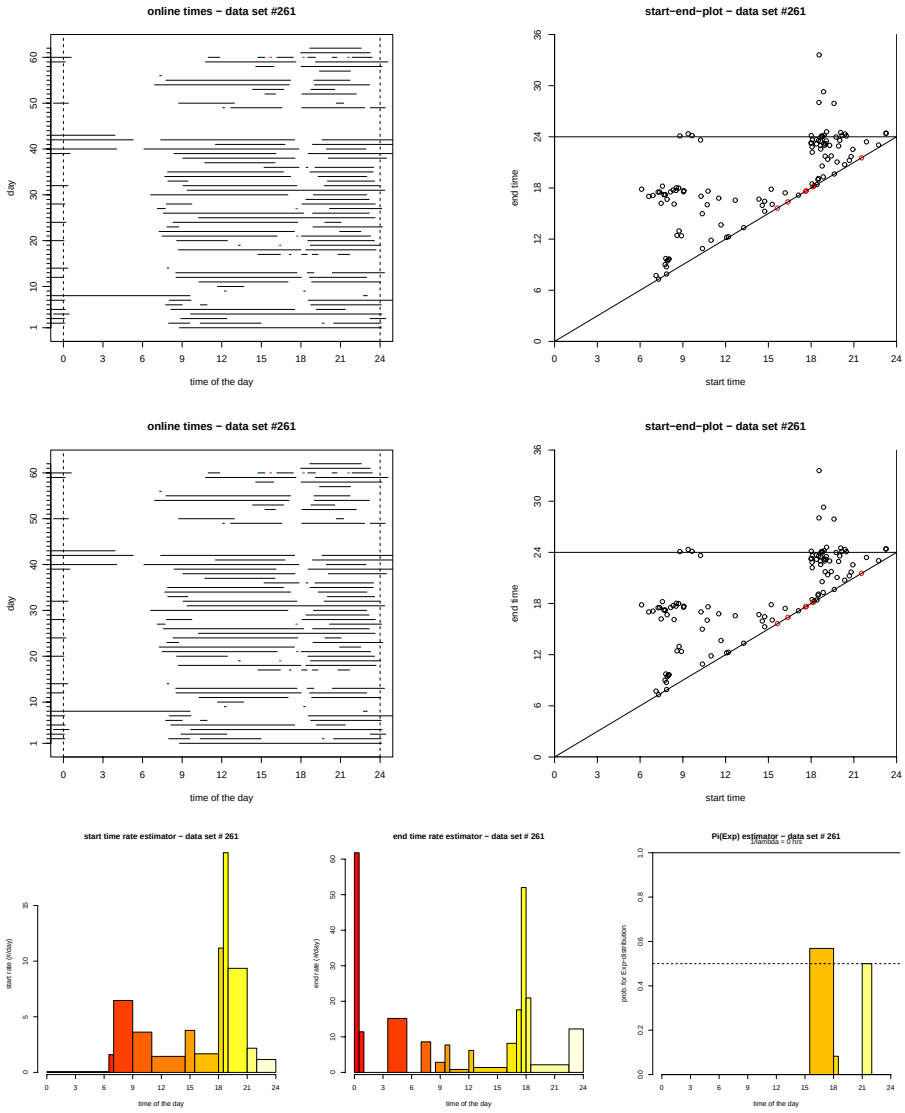

Figure 4: Original data, simulated data (XXX), and estimated parameters for a second user. Lunch breaks are recognized.

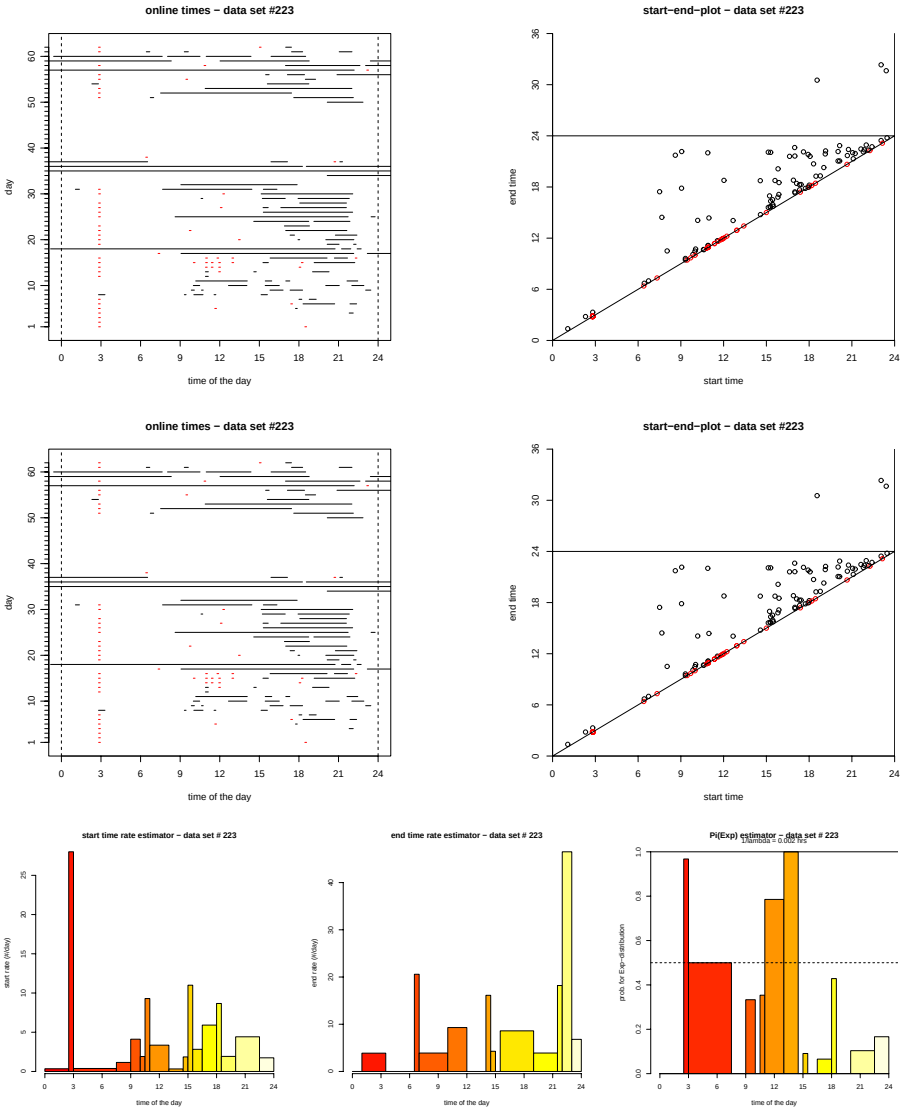

Figure 5: Original data, simulated data (XXX), and estimated parameters for a third user with a very irregular behavior. The algorithm managed to capture the caonnections regularly appearing at 3 pm.