Continuous Time Bayesian Networks for Inferring Users' Presence and Activities with Extensions for Modeling and Evaluation

Uri Nodelman[1] **Eric Horvitz**

Microsoft Research
One Microsoft Way
Redmond, WA 98052 USA

nodelman@cs.stanford.edu horvitz@microsoft.com

Microsoft Research Technical Report
MSR-TR-2003-97

December 2003

Abstract

Continuous time Bayesian networks (CTBNs) represent structured stochastic processes that evolve over continuous time. The methodology is based on earlier work on homogenous Markov processes, extended to capture dependencies among variables representing continuous time processes. We have worked to apply CTBNs to the challenge of reasoning about users' presence and availability over time. As part of this research, we extended the methodology of CTBNs to allow a large class of phase distributions to be used in place of the exponential distribution that had previously been used. We also formulate and examine the use of classes of cost functions for evaluating the performance of CTBN models in the real world. The cost functions represent the loss associated with inaccuracies in the predicted time and states of transitions of processes.

1 Introduction

Continuous time Bayesian networks (CTBNs) describe structured stochastic processes with finitely many states that evolve over continuous time. A CTBN is a directed (possibly cyclic) dependency graph over a set of variables, each of which represents a finite state, continuous time Markov process whose transition model is a function of its parents. The representation, and inference and learning methodologies, are described in earlier papers (Nodelman et. al., 2002; Nodelman et al., 2003). We will not delve into details of CTBNs in this report. We refer readers to technical details of the earlier work.

[1] The first author is at the Computer Science Department, Stanford University, Stanford, CA 94305. This work was performed while he was an intern at Microsoft Research, July-August, 2003.

Continuous time Bayesian networks (CTBNs) represent structured stochastic processes that evolve over continuous time. The methodology is based on earlier work in the decision science community on homogenous Markov processes. However, with CTBNs, we enrich homogenous Markov processes to capture dependencies among the intensity matrices of sets of interacting variables representing different continuous time processes. With CTBNs, we represent conditional intensity matrices to represent influences among stochastic processes. Figure 1 shows a CTBN that represents the type of sample problem that has been at the focus of some earlier research on CTBNs. In this case, we are modeling the relationships among the stochastic processes representing transitions that influence a target inference about the concentration of a pharmaceutical agent at a patient's joints.

In our work, we sought to apply CTBNs to the challenge of reasoning about users' presence and availability over time. As part of this research, we extended the methodology of CTBNs beyond earlier work limited to representing stochastic processes with exponential distributions, so as to allow a large class of *phase distributions*. We have also formulated and examined the use of multiple classes of cost functions that represent the losses associated with inaccuracies in the predicted time and predicted transitions of processes.

We review our study of CTBNs for inferring presence and availability, discuss extensions motivated by the project, and review new observations, challenges, and questions that arose during the course of this effort.

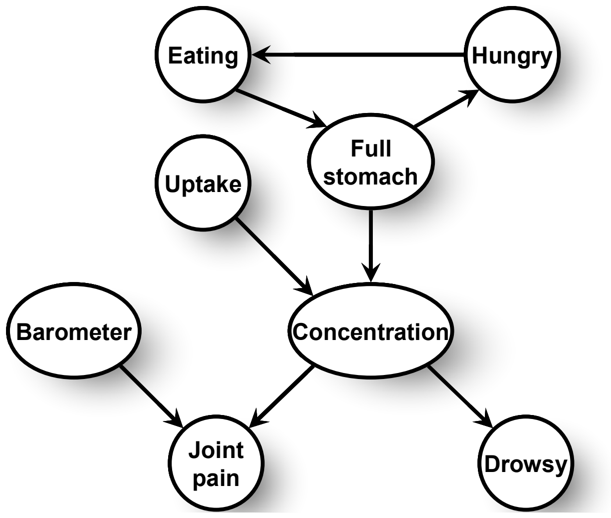


Figure 1. A CTBN representing the changes in a concentration of an analgesic at a joint over time, and the associated diminishment of joint pain as a function of the interaction of multiple stochastic processes.

2 Background on Coordinate

Coordinate is a research project centering on machine learning to build predictive models for forecasting a user's presence and availability (Horvitz, et al. 2002). Prior work on Coordinate explores the use of dynamic Bayesian networks (DBNs) and dynamically constructed single-stage models that represent distinctions about time in the definitions of multiple variables. In the latter case, models are generated dynamically at query time, by (1) identifying sets of matching data, (2) performing structure search with a Bayesian score to build predictive models, and (3) doing inference with one or more models.

Figure 2 shows a high-level schematic of Coordinate. The system monitors multiple channels of activity and maintains a database of transitions and contextual feature, and performs learning and inference in both an offline mode and in real-time, in response to queries, depending on the configuration and inferences. Figure 3 displays a sample output of Coordinate, showing predictions available on a snapshot or "presence palette" of the predicted times until a user will be present and online at home and office locations, the time until a user will be in a situation that provides videoconferencing capabilities, and the time until a user will review email. Viewers with privileges to see this information can set different thresholds so as to display the times that transitions will occur with different probabilities. For the presence palette displayed in Figure 3, the time until transitions will occur with a 0.90 probability are displayed.

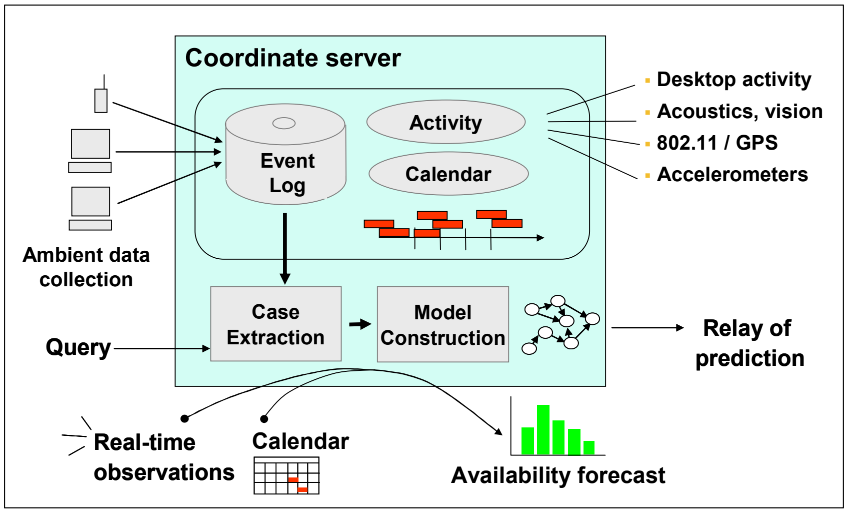
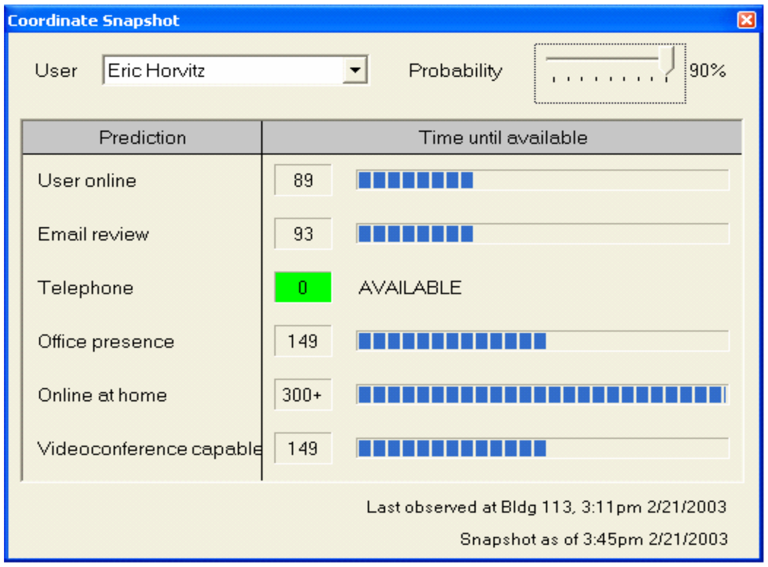


Figure 2. Overview of the architecture and operation of Coordinate.



Coordinate Snapshot

User   Eric Horvitz   Probability   90%

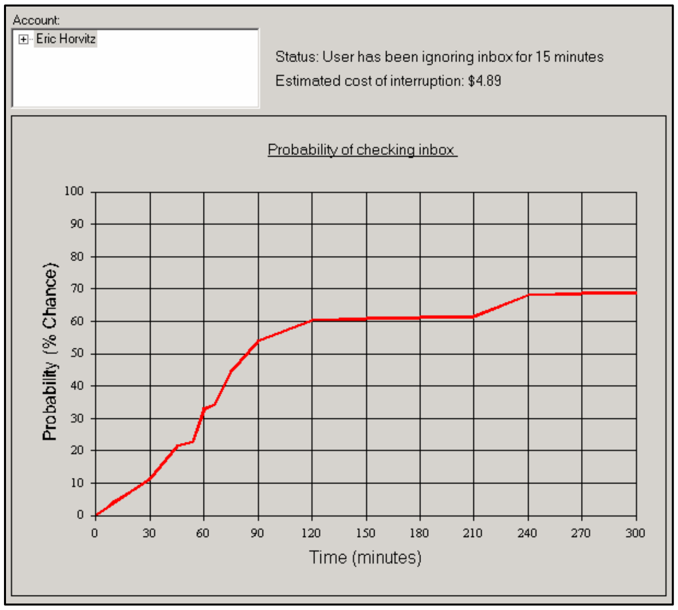
Prediction	Time until available
User online	89
Email review	93
Telephone	0 AVAILABLE
Office presence	149
Online at home	300+
Videoconference capable	149

Last observed at Bldg 113, 3:11pm 2/21/2003

Snapshot as of 3:45pm 2/21/2003

Figure 3. Coordinate presence palette, as displayed in a “snapshot” view, providing viewers with information about the predicted times until a user will be available at different locations and with different communication capabilities. In this case, the view has been set to show the inferred times until a user will be available with a 0.9 probability. As indicated by the green square, the user is currently available by telephone.

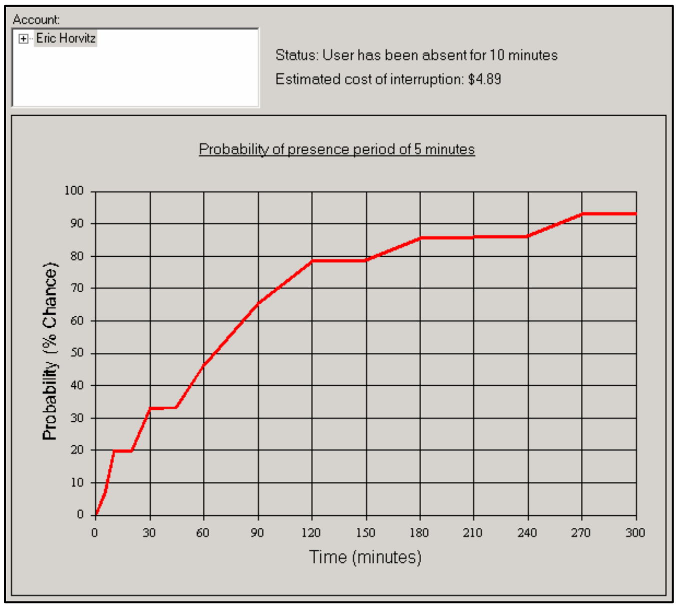
Coordinate also makes available detailed views of inferred probability distributions and cumulative probability distributions over the predicted times until transitions of interest will occur. Figure 4 displays the details of the inferred cumulative distribution about the time until a user will check their inbox. Figure 5 displays the details of the time until a user, who is not at the moment not present in their office, will return to their office, and will be in their office for at least 5 minutes. We refer readers to (Horvitz, et al. 2002) for additional details about Coordinate.



Status: User has been ignoring inbox for 15 minutes

Estimated cost of interruption: $4.89

Figure 4. Detailed view of inferred cumulative probability distribution over the predicted time until a user will check their email.



Status: User has been absent for 10 minutes

Estimated cost of interruption: $4.89

Figure 5. Detailed view of inferred cumulative probability distribution over the predicted time until a user will return to their office and remain for at least 5 minutes.

3 CTBNs for Modeling Presence and Availability

We now turn to work on the use of CTBNs for tackling Coordinate challenges. We focused on two modeling two classes of related transitions: the presence of people on their computers and, while they are present on their computers, the application that is in focus, or "on top," *i.e.,* the application in which they are working currently (e.g., email, word processing, web browsing, etc.). The model has, at its core, two variables: *Presence* and *ApplicationOnTop*. The CTBN graph describing the overall process encompassing both variables is displayed in Figure 6.

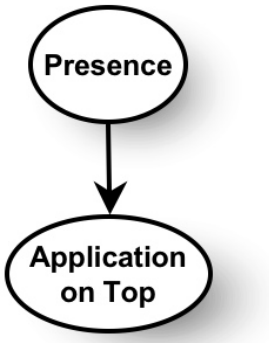


Figure 6. A CTBN for predicting the likelihood of computing applications becoming the top active application for a user.

With such a model, we seek to predict the future values of these variables, in order to answer questions such as:

1) When do we next expect this person to be present on their computer?
2) When do we expect this person to next use Outlook? (or any other particular application?)
3) How much time do we expect the person to remain on their computer?
4) How much time before we expect the person to switch to a different application?

More information is needed to make this model as accurate. For example, the pattern of presence of a person on a computer often varies depending on whether it is a weekday or the weekend. So we would like to add other variables. For example, we can add other another variable to the network describing whether the activity of interest is occurring on a weekday or weekend, as displayed in Figure 7.

This CTBN graph structure encodes an independence assumption, namely, if given the pattern of presence of a person on the computer, then the pattern of what application is on top does not depend on whether or not it is a weekday. If this assumption is untrue, then an additional arc from *Weekday* directly to *ApplicationOnTop* should be added.

In general, we will learn both structure and parameters of these networks from data, so we will not need to make such decisions ourselves. After learning parameters for this CTBN, we can use it to plot the behavior of the system. Coordinate collects data from multiple computers, so we can build models that reason over application for all computers owned by a user, or on a single specific computing device.

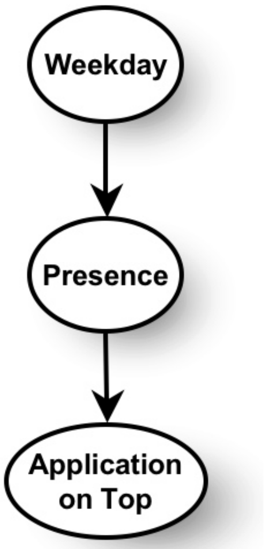


Figure 7. A CTBN for predicting the likelihood of computing applications becoming the top active application for a user, extended to include information about the period of time associated with the target inference.

Figure 8 shows the output of a CTBN learned from Coordinate data about the time until Outlook will return to being the top, active application for the situation where Outlook is not currently in focus. We plot the the likelihood that a user will bring Outlook to the top and become active with the application as a function of time and the application that was in focus at time zero. For this user, a transition to Outlook is expected to occur most quickly for the case where the user is in currently at the top level of the computer, working in the shell, followed by the case where the user is currently reading a .pdf file, and then working with Microsoft Word. Significantly later transitions to Outlook are noted for the case where the user is currently browsing or reading materials on the Web (user in Internet Explorer), and even later for the case where the user is using Microsoft Powerpoint.

We can build more complex CTBNs. For example, if we are particularly interested in when the person will next use Outlook, we can add a variable that keeps track of how long it has been since the user last used Outlook. This extended model, with these additional variables, is displayed in Figure 9.

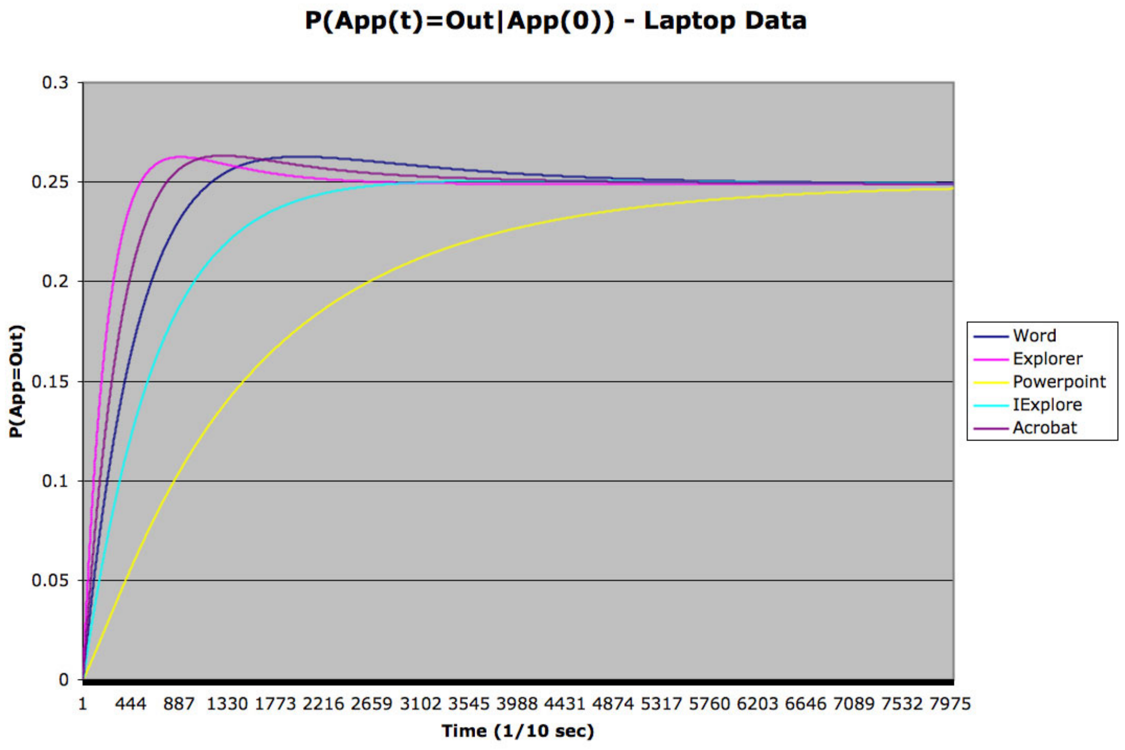


Figure 8. Inferences from a CTBN learned from Coordinate data about the time until a transition is made to Outlook, given that other applications are active at time 0, based on data from a single computing device.

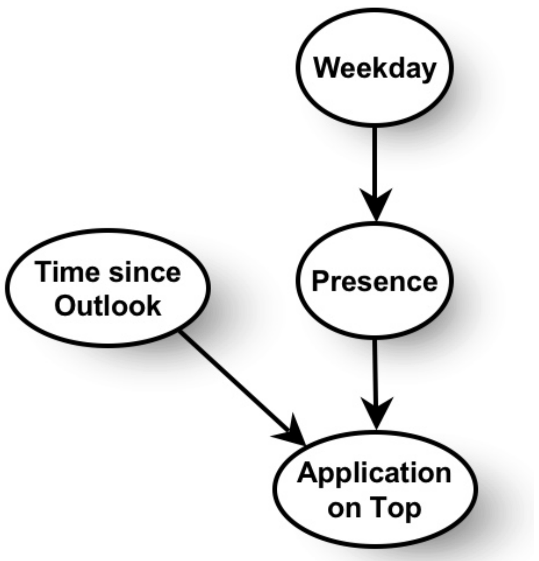


Figure 9. Extending the model to consider the amount of time that has transpired since a user last used Outlook.

4 CTBN Extension: Phase Distributions for Duration in a State

The challenges of the Coordinate project motivated our pursuit of generalizations of CTBNs to use distributions beyond the exponential. To date, CTBNs have employed exponential distributions over when the next transition will occur, given the complete state of the system at the initial time. As part of this project, we have developed an extension of the regular CTBN framework that allows the large class of *phase distributions* to be used in place of the exponential.

Formally, a phase distribution is the distribution over the time to absorption to an absorbing state of a continuous time Markov process. Phase distributions can be made that have almost any density (approximately), though more complex distributions will generally require more parameters. (The Markov process that defines the phase distribution is made of several states or *phases* each of which can be parameterized independently.) Erlang distributions form a simple but very useful subclass of phase distributions. Figure 10 shows graphs of the density function for a few examples.

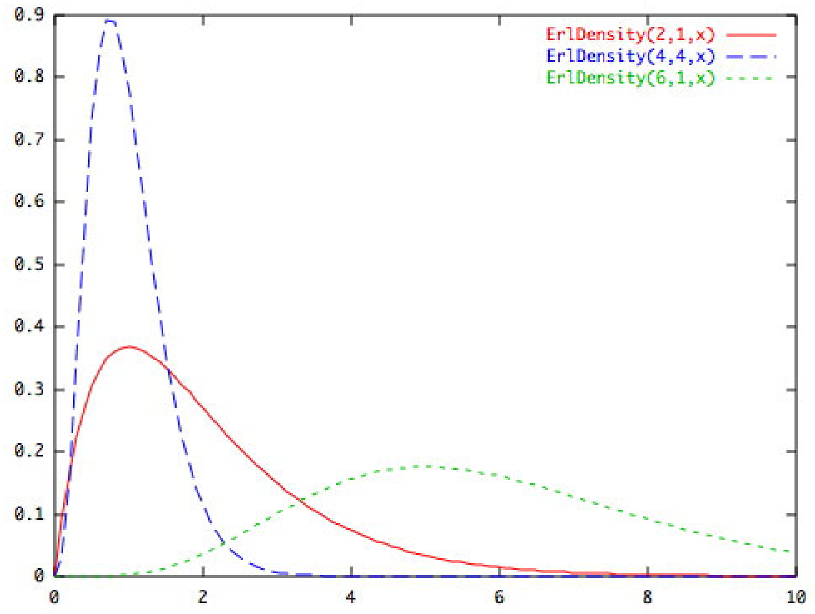


Figure 10. Graphs of density functions of Erlang distributions for sample parameters.

One of the advantages of Erlang distributions is that they have relatively few parameters. In fact, if one has decided on a fixed number of phases, then there is only one parameter which determines the mean of the distribution.

In general, a complex duration CTBN is formed by allowing each state of each variable to be represented by some number of *substates*. If value x of variable X is represented by 3 substates x_1, x_2, and x_3, then X has value x if and only if it is in any of these substates.

Given the representation of the transitions between the substates written as an intensity matrix, we can write the distribution over when the variable leaves the state in closed form with a matrix exponential. For the class of Erlang distributions, this can simplified further.

We have explored the use of Erlang distributions, on challenges with building and using CTBN models with Coordinate data for the task of inferring transition times on presence and availability.

5 Validation for CTBN Models

In addition to studying Erlang distributions, we have also extended efforts on a methodology for evaluating the performance of CTBN models. Comparing CTBNs to other temporal models for validation is generally difficult. Most work in machine learning has employed likelihood models, *e.g.,* log likelihood, to score the performance of models. However, likelihood measures for DBNs and CTBNs are not directly comparable. We developed an approach to comparing the performance of CTBNs and other temporal models by using expected loss. We define a class of loss functions that have the following properties:

1. The *total loss for a trajectory* given a model as the sum of local losses, one for each transition of interest in the trajectory.
2. The *local loss* as a measure of how well that particular transition is predicted by the model.
3. The loss function is zero at the point of transition and nonnegative to the left and the right of the transition time.
4. The basic shape of the loss function may be linear, quadratic, exponential, etc., with independent parameters for the left and right portions of the loss function.

For example, we can consider a linear loss function:

lin<alpha, beta, t'>(t) = alpha(t' – t) for t < t'
and beta(t – t') otherwise,

where t' is the actual time of transition and t is the predicted time of transition. The parameters alpha and beta are slopes for the 2 linear portions of the loss function.

Expected loss can then be computed as the expected value of the loss function given the distribution over when that transition will take place according to the model. We can include in the loss a penalty term if the transition is not predicted by the model.

Figure 11 displays a graph of the lin<1,1,2>(t) loss function with the CTBN density, employing an exponential form, over when that transition will occur in red. In this case, the exponential density has been chosen such that the mean is equal to the actual time of transition, t'=2. The green function shows the product of the loss function and the density so that the expected loss is the area under the curve.

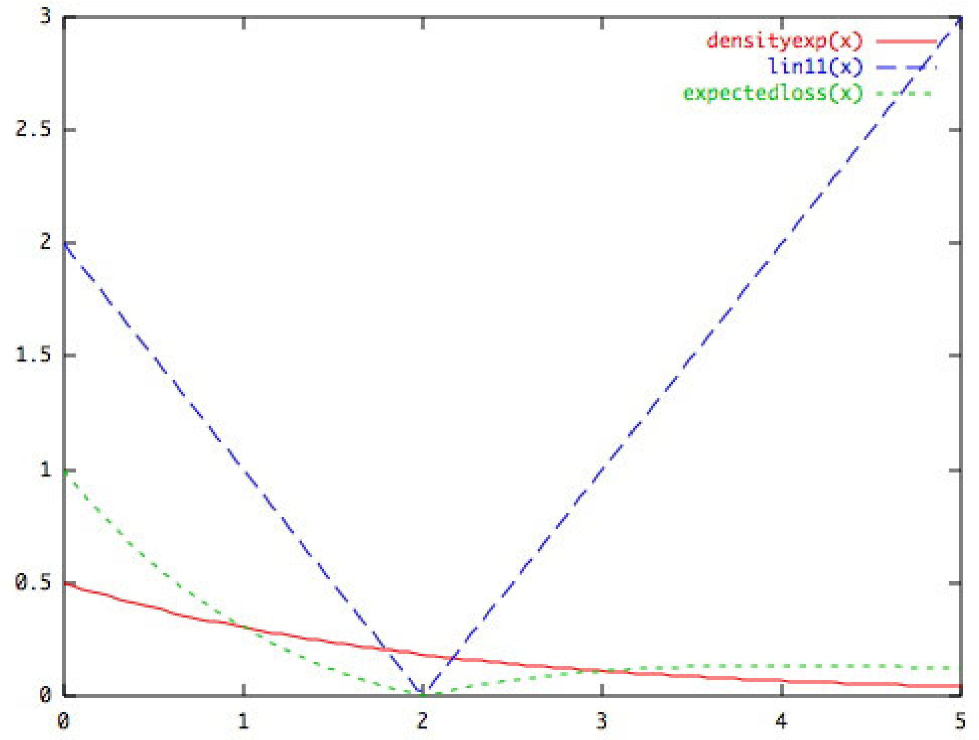


Figure 11. Graphs of density a graph of the linear loss function, lin<1,1,2>(t). The density over the transition time computed by a CTBN is indicated in red. In this case, the exponential density has been chosen such that the mean is equal to the actual time of transition, t'=2. The green function shows the product of the loss function and the density, so that the expected loss is the area under the curve.

Given the above linear loss function, it is possible to compute the expected loss in closed form for a CTBN as a function of the relevant intensity parameter q and the actual time of transition t as follows:

CTBNExpectedLoss(q,t) = (alpha + beta)/q * exp(-qt) + alpha(t-1/q)

Given this equation, we can plot the expected loss surface (z-axis) as a function of q (y-axis, from 0 to 2) and t (x-axis from 0 to 10). This is for the case where the actual time of transition is 2. Note that the if q = 0.5 then the expected time of transition is 2. We show this loss surface in Figure 12.

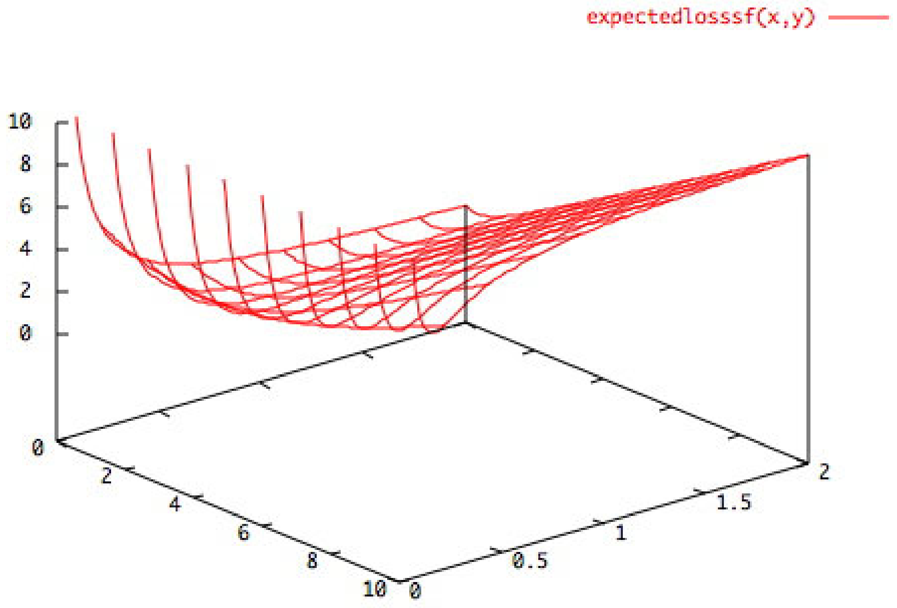


Figure 12. Expected loss surface (z-axis) as a function of q (y-axis, from 0 to 2) and t (x-axis from 0 to 10) for the case where the actual time of transition is 2.

6 Application to Coordinate Data

We have performed a number of experiments using expected loss to compare DBNs with different time granularities to regular CTBNs and Erlang-based CTBNs.   For Erlang CTBNs we used 2 phase and 3 phase Erlangs. In general, the results showed that the performance of the models, from best to worst, is: Erlang-2 CTBNs > DBNs > Regular CTBNs > Erlang-3 CTBNs.

However, these results may be somewhat misleading because we calculated expected loss from models that were learned with Bayesian scoring and Bayesian parameters. These parameters are not optimized for minimizing expected loss. In fact, we found that increasing the amount of data used for learning often hurt performance.

In order to investigate these phenomena in more detail, we examined the empirical distributions of the duration times of different computing applications being in focus. Figures 13 through 16 show histograms of these durations. We show histograms for durations of Internet Explorer, Outlook, PhotoEditor, and Word. In order to see the pattern more clearly, these histograms are truncated after 1 minute even though there are durations that greatly exceed 1 minute.

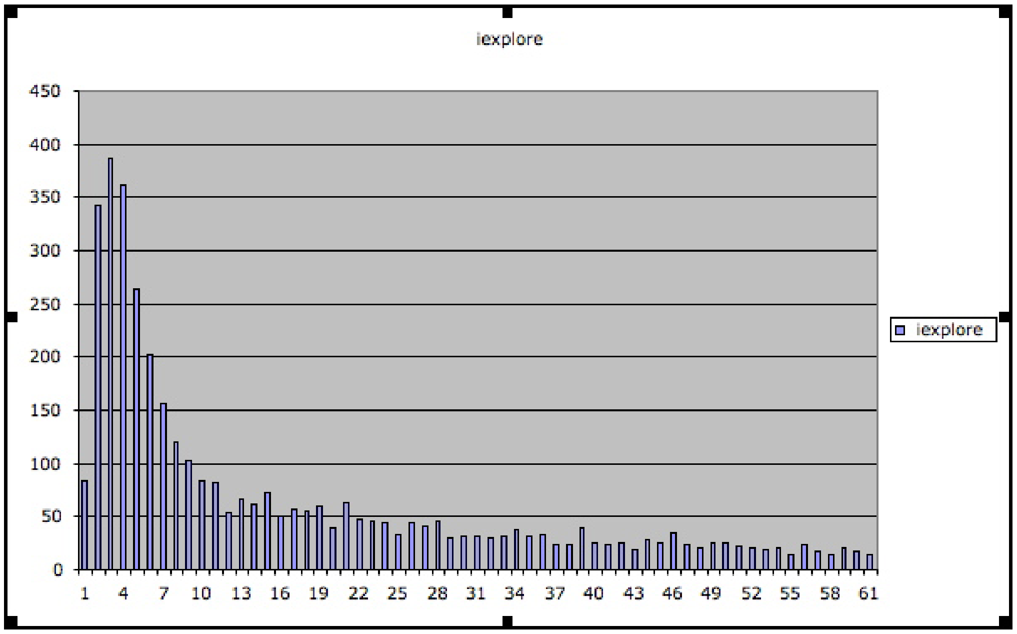


Figure 13. Histogram for durations of the time a user spends in Internet Explorer.

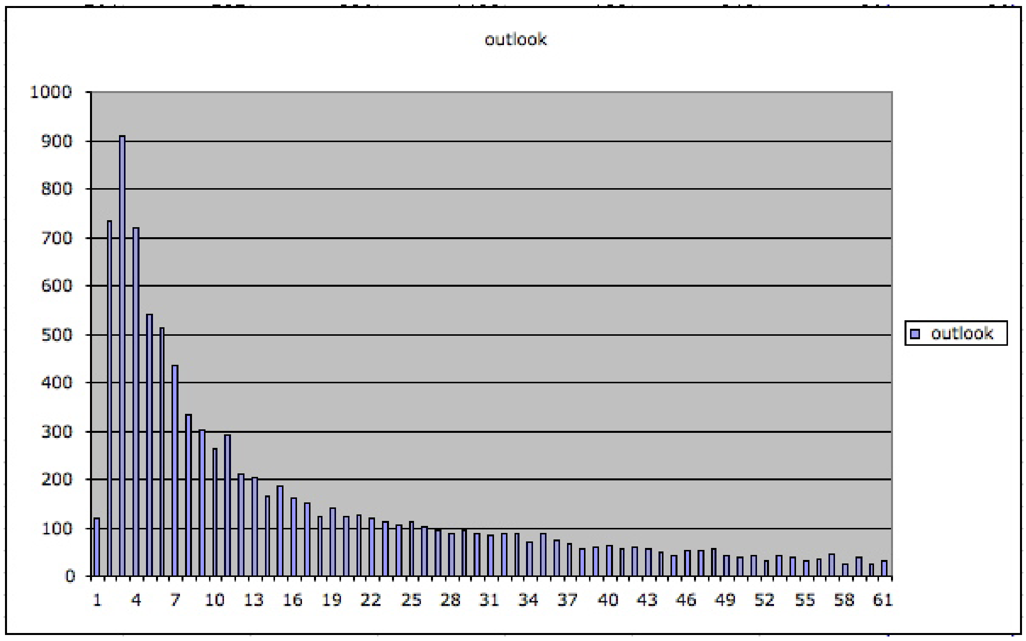


Figure 14. Histogram for durations of the time a user spends in Outlook.

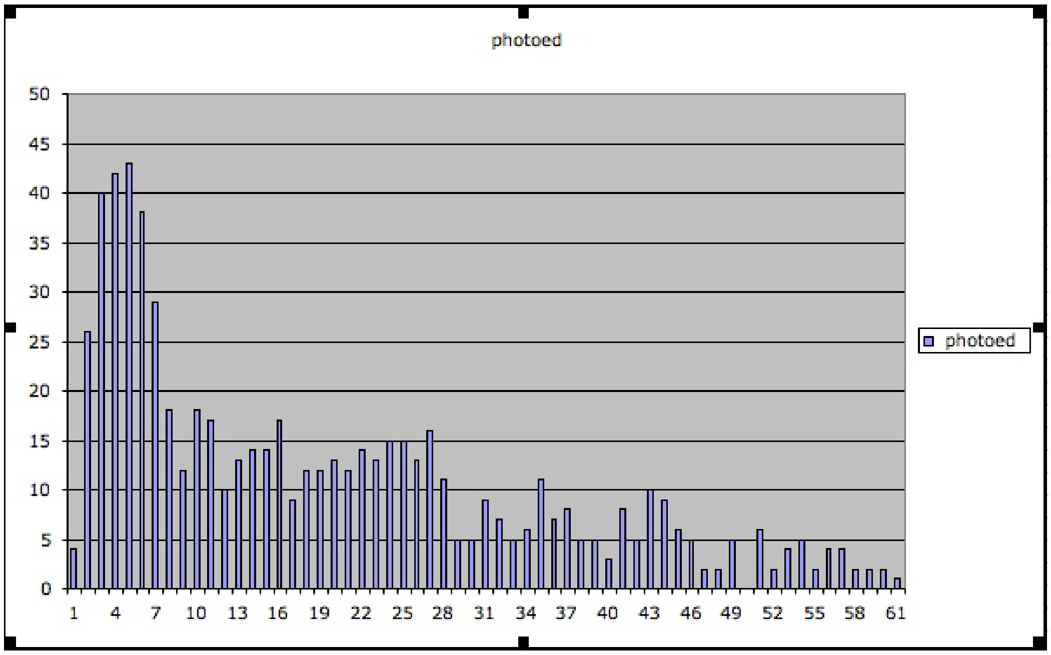


Figure 15. Histogram for durations of the time a user spends in PhotoEditor.

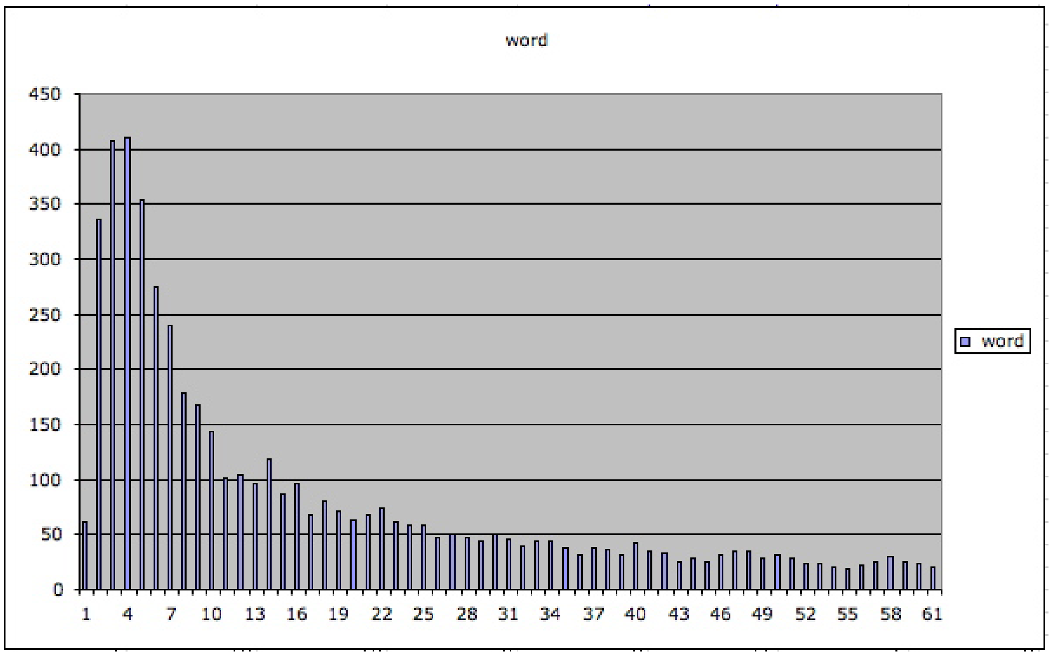


Figure 16. Histogram for durations of the time a user spends in MS Word.

We point out two important features of these distributions: (1) The distributions are shaped more like Erlang distributions than exponential (or geometric) distributions. (2) The distributions have long tails such that the mean is much greater than the mode.

(1) may help explain why Erlang-2 distributions did better than DBNs and regular CTBNs.[2] (2) may help explain why DBNs (with geometric distributions) did better than regular CTBNs (with exponential distributions). Even though the geometric distribution can be seen as a discrete analog to an exponential distribution, there are some important systematic differences. Namely, if the distributions have the same mean, the geometric puts more probability mass on transitions that occur before the mean and less on transitions that occur after the mean compared to the exponential distribution. This effect is more pronounced with larger time granularities; as the time granularity approaches zero, the geometric distribution becomes a better and better approximation of the exponential distribution. We can see this from graph in Figure 17, which shows an exponential distribution and geometric distributions for different time granularities, all having a mean of 2.

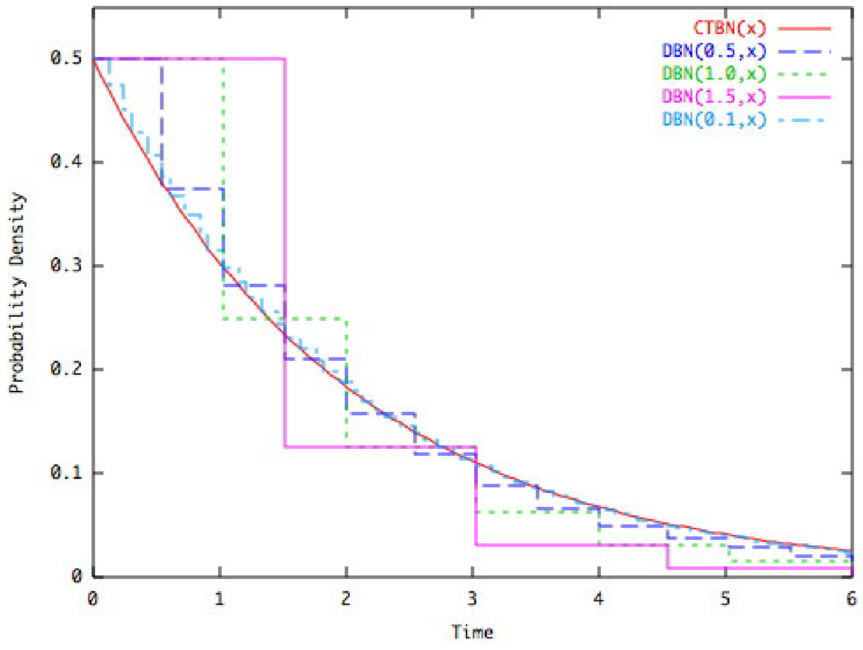


Figure 17. An exponential distribution and geometric distributions for different time granularities, all having a mean of 2.

Given that the Bayesian parameters tend, with enough data, to match the mean of the empirical distribution, and that we are in a situation where more durations are smaller than the mean, we expect the DBN to do better because of its bias towards shorter durations. This explanation is supported by the observation that DBNs with larger time granularities tended to do better in the experiments than DBNs with smaller time granularities. In general, we would expect that as the time granularity approaches zero, the performance of the DBN would match the CTBN. We can show this more clearly with the graph displayed in Figure 18, showing the expected loss versus transition time for CTBNs and DBNs of different time granularities. The mean transition time for all distributions is 2. Note that if the actual transition time is less than the mean, the expected loss is smallest for the DBN with the largest time granularity.

[2] The Erlang-3 distributions are more peaked and thus more sensitive to the parameter chosen. The fact that they did even worse seems related to the fact that the parameters were not chosen to minimize expected loss.

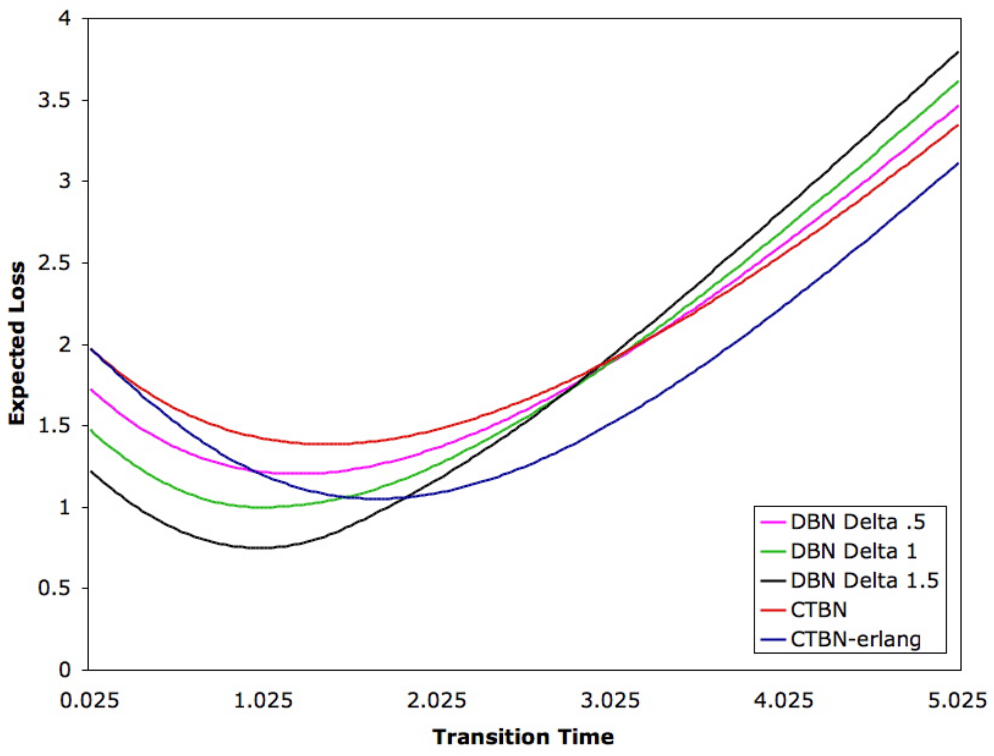


Figure 18. The expected loss versus transition time for CTBNs and DBNs of different time granularities, with mean transition time for all distributions equal to 2.

7 Conclusion

We have reviewed work on applying CTBNs to the challenge of reasoning about users' presence and availability. As part of this research, we extended CTBNs with the use of phase distributions, thus extending CTBNs beyond a reliance on exponential distributions. We have also moved beyond the use of likelihood as a measure of model performance, and formulated and examined the use of cost functions that represent the losses associated with inaccuracies in the predicted time and predicted transitions of models of stochastic processes. Finally, we explored questions about the interplay between models of cost and learning procedures, in a comparison of variants of CTBNs and DBNs.

Acknowledgments

We thank Paul Koch and Johnson Apacible for assistance with accessing Coordinate data.

References

(Horvitz, *et al.*, 2002)
Horvitz, E., Koch, P., Kadie, C.M., and Jacobs, A. Coordinate: Probabilistic Forecasting of Presence and Availability. *Proceedings of the Eighteenth Conference on Uncertainty and Artificial Intelligence*, Edmonton, Canada, July 2002, pp. 224-233.
http://research.microsoft.com/~horvitz/Coordinate.htm

(Nodelman, *et al.,* 2002)
Nodelman, U., Shelton, C. R., and Koller, D. (2002). Continuous Time Bayesian Networks. *Proceedings of the Eighteenth International Conference on Uncertainty in Artificial Intelligence*, July 2002, pp. 378-387.
http://ai.stanford.edu/~nodelman/papers/ctbn.pdf

(Nodelman, *et al.,* 2003)
Nodelman, U., Shelton, C. R., and Koller, D. (2003). Learning Continuous Time Bayesian Networks. *Proceedings of the Nineteenth International Conference on Uncertainty in Artificial Intelligence,* August 2003*,* pp. 451-458.
http://ai.stanford.edu/~nodelman/papers/learn-ctbn.pdf