Modeling Travel and Activity Routines using Hybrid Dynamic Mixed Networks

Vibhav Gogate

Donald Bren School of Information and Computer Science
University of California, Irvine
Irvine, CA 92967

Rina Dechter

Donald Bren School of Information and Computer Science
University of California, Irvine
Irvine, CA 92967
tel: 949-824-6556
email: dechter@ics.uci.edu

Bozhena Bidyuk

Donald Bren School of Information and Computer Science
University of California, Irvine
Irvine, CA 92967

Craig Rindt

Institute of Transportation Studies
University of California, Irvine
Irvine, CA 92967
tel: 949-824-6544
email: crindt@uci.edu

James Marca

Institute of Transportation Studies
University of California, Irvine
Irvine, CA 92967
tel: 949-824-6544
fax: 949-824-8385
email: jmarca@uci.edu

November 21, 2005

Keywords: activity modeling, Bayesian networks, GPS

Wordcount: 6150 + 3 figures + 4 tables = 7900

Modeling Travel and Activity Routines using Hybrid Dynamic Mixed Networks

Vibhav Gogate , Rina Dechter , Bozhena Bidyuk , Craig Rindt , James Marca

Abstract. This paper describes a method for storing, modeling, and processing an individual's travel history collected with a GPS enabled device. The technique uses a general framework called Hybrid Dynamic Mixed Networks (HDMNs), which are Hybrid Dynamic Bayesian Networks that allow representation of discrete deterministic information in the form of constraints. We use this framework to model a person's travel activity over time and to infer likely destinations and routes given information about the current trip. We present a preliminary empirical evaluation demonstrating the effectiveness of our modeling framework and algorithms using three variants of the activity model. This research will improve the quality of activity surveys based on electronic data collection methods, as well as improve the usefulness and effectiveness of in-vehicle and hand-held devices for daily activity planning and rescheduling.

INTRODUCTION

The increasing proliferation of powerful mobile computing devices is already creating an explosion of high resolution, real-time data about the travel and activity behavior. To this point, use of this data has been largely focused on aggregative techniques that mirror existing approaches to estimating the state of the system. For instance, mobile phone operators are beginning to allow data about their subscribers' locations to be used for modeling traffic flow as a replacement for costly and unreliable infrastructure-based sensors. There has also been a lot of recent activity to leverage vehicle to vehicle communications to propagate network performance information among the vehicle collective.

In our view, however, these mobile nodes can potentially serve the broader purpose of archiving this newly available travel behavior data in individual travel history databases. A personal travel assistant application could leverge such a database to help the traveler plan or replan their daily travel, taking into account prior travel histories as well as real-time information to anticipate the user's planned behavior and provide active support by identifying disruptions that will impact the traveler. This should improve adoption of the technology by reducing the burden on the user to manually define current travel goals, and therefore increase overall efficiency of the system (from the users' perspectives) by improving travelers' responsiveness to disruptions.

Furthermore, if a small sample of the traveling public agreed to make their personal travel history databases publicly available, transportation management systems could incorporate predictions of current individual travel demands into broader dynamic traffic assignment (DTA) systems. This would improve real-time demand estimates and lead to better efficiency through improved control.

This paper explores one technique for building a personal travel history database that can be leveraged in a broad class of traveler support and systemic modeling applications. Assume an individual carries a portable computing device, such as a cell phone, that has moderate processing power, relatively limited memory for storing data, and a GPS receiver. We seek a system that can learn the individual's behavioral patterns and answer queries about future behavior with sufficient confidence to support the personal and system-wide applications mentioned above.

While the GPS provides a time-series of highly accurate geospatial coordinates, the volume of data is far too great to be archived directly on embedded devices. Furthermore, analysis of travelers' movements requires mapping the raw GPS data to a network representation of the transportation system. Complicating this map matching problem are errors in both the GPS data and in the underlying travel network representation. Any system wishing to describe travel behavior analytically must compensate for or correct these errors. A second challenge is to efficiently store a traveler's prior travel history in a manner that facilitates accessing the history for later analysis. As a travel history builds, the complete enumeration of data will further tax the limited resources available in embedded systems. The third challenge is to provide an efficient lookup mechanism so that queries can be made on platforms with limited computational power.

Various theory-driven modeling approaches may be suitable for a subset of these problems. For instance, there are myriad approaches for representing an individual's travel behavior over time. Most of these are cross-sectional approaches that combine multiple observations to create general models of behavior that are applied to model any individual. Application of discrete choice modeling to longitudinal modeling is less common. A recent example is the work carried out by Srinivasan and Mahmassani on the dynamic kernel logit (DKL) and multinomial probit (MNP) models. These efforts are explicitly targeted at modeling dynamic discrete choice data, such as departure time switching and route choice switching behaviors *(20)*. A major difficulty with any discrete choice formulation, and dynamic discrete choice in particular, is that it must represent a series of discrete choices explicitly. For instance, it is very difficult to represent raw route choice behavior using a discrete choice formulation because the choices, and the order in which they are made, must be specified *a priori*. As such, it is easier to model choices such as route switching, rather than actual route choice behavior. It is difficult to imagine a tractable discrete choice forumulation that can effectively build an on-line model of an individual's travel behavior (a joint model of route, destination, or activity pattern choice) given longitudinal GPS observations and reference map data that can be used to predict future behavior given a few observations from the current trip.

Models of sequential choice behavior are available in the literature, particularly in the area of activity modeling, which includes both decision theoretic approaches and techniques relying on artificial intelligence. rec's 1995 household activity pattern problem (HAPP) represents the decision theoretic extreme by forumulating a household's daily activity and travel behavior decision problem as a mathematical program that optimizes some generalized utility function. The problem of calibrating the utility function from observed behavior, however, remains a research challenge *(3)*. *(2)* took a different approach to the same problem with their *Albatross* model. The model system predicts the activity of a household based upon decision rules empirically calibrated to best match observed behavior according to a pattern similarity metric. *Albatross* has shown promise as a policy tool. More recent work to incorporate theory-based models of learning and adaptation in travel behavior also show promise *(4)* for capturing the full range of travel and activity behavior dynamics.

For all of their elegance and rigor, however, it is difficult to imagine a on-line versions of any of these models that can learn a single individual's behavioral patterns and predict that individual's current behavior based solely upon GPS observations of behavior in progress. Our approach is to construct a statistical model of an individual's travel behavior based on observed behavior in time and space to reduce the extensive GPS data available to modern mobile devices into a form that could potentially be deployed on those devices desite their limited computational and memory resources. By adopting a statistical approach, rather than a more theory-driven approach, we gain flexibility that makes the system more adaptable to multiple applications. While this may hamper its use in policy applications that need to predict behaviors that may not yet have been observed, our focus is on supporting operational applications in which travelers' behaviors are more likely to be based upon routines that have been observed rather than on exploratory behaviors that aren't present in the data.

Given a reference map of the transportation network and a set of longitudinal GPS data recording an individual's travel behavior, our approach uses Hybrid Dynamic Mixed Networks (HDMN) to construct a queryable model of a individual traveler's destination and route choice behavior. Queries are formed by providing a small set of GPS data (e.g., recording the beginning of a new trip) and return predictions about likely destination and route for that trip. The framework is amenable to on-line application though we don't demonstrate that capability here.

In the next section, we discuss preliminaries and introduce our modeling framework. We then describe two approximate inference algorithms for processing HDMN queries: an Expectation Propagation type and a Particle Filtering type. Subsequently, we describe the transportation modeling approach and present preliminary empirical results on how effectively a model is learned and how accurately its predictions are given several models and a few variants of the relevant algorithms.

PRELIMINARIES AND DEFINITIONS

Modeling sequential real-life domains often requires the ability to represent both probabilistic and deterministic information. Graphical models have been used for this purpose to visualize and encode the relationships between variables. While they are easily described for systems with three or four variables, their strengths become apparent when many variables are involved, as is the case with travel routing and planning. Conventional graphic models such as Bayesian Networks, however, are too restrictive to represent anything meaningful in the travel behavior domain. In particular, we need to be able to model behavior that is:

- dynamic (because travel behavior occurs over time);
- represented by a combination of variables that are both discrete (e.g., destination chosen, link used, etc.) and continuous (e.g., position in space); and
- bounded by known constraints (e.g., time available for an activity).

In this section we describe Hybrid Dynamic Mixed Networks, which meet all of these requirements, as an extension to the more familiar Bayesian Networks. *Bayesian Networks* are graphical models that specify the probabilistic dependencies between variables. Typically, Bayesian networks contain just discrete variables. To build a model with both discrete and continuous variables we must use a *Hybrid* Bayesian Network.

Formally, *Hybrid Bayesian Networks (HBN) (15)* are graphical models defined by a tuple $\mathcal{B} = (X, G, P)$, where X is the set of variables, G is a directed acyclic graph whose nodes correspond to the variables, and $P = \{P_1, ..., P_n\}$ is a set of conditional probability distributions (CPDs). Given variable x_i and its parents in the graph $pa(x_i)$, the conditional probability P_i is defined as $P_i = P(x_i|pa(x_i))$. In a hybrid network, X is partitioned into discrete and continuous variables $X = \Gamma \bigcup \Delta$, respectively. Further, the graph structure G is restricted such that continuous variables cannot have discrete variables as their child nodes. The conditional distribution of continuous variables are given by a linear Gaussian model: $P(x_i|I = i, Z = z) = N(\alpha(i) + \beta(i) * z, \gamma(i)) x_i \in \Gamma$ where Z and I are the set of continuous and discrete parents of x_i, respectively and $N(\mu, \sigma)$ is a multi-variate normal distribution. The network represents a joint distribution over all its variables given by a product of all its CPDs.

To handle constraints on behavior in a graphical model, we can use a *Constraint Network (6)*, which is a graphical model $\mathcal{R} = (X, D, C)$, where $X = \{x_1, \ldots, x_n\}$ is the set of variables, $D = \{D_1, \ldots, D_n\}$ is their respective discrete domains and $C = \{C_1, C_2, \ldots, C_m\}$ is the set of constraints. Each constraint C_i is a relation R_i defined over a subset of the variables $S_i \subseteq X$ and denotes the combination of values that can be assigned simultaneously. A *solution* is an assignment of values to all the variables such that no constraint is violated. The primary query is to decide if the constraint network is consistent and if so find one or all solutions.

The recently proposed Mixed Network framework *(9)* for augmenting Bayesian Networks with constraints, can immediately be applied to HBNs yielding the *Hybrid Mixed Networks (HMNs)*. Formally, given a HBN $\mathcal{B} = (X, G, P)$ that expresses the joint probability $P_{\mathcal{B}}$ and given a constraint network $\mathcal{R} = (X,D,C)$ that expresses a set of solutions ρ, an HMN is a pair $\mathcal{M} = (\mathcal{B},\mathcal{R})$. The discrete variables and their domains are shared by $\mathcal{B}$ and $\mathcal{R}$ and the relationships are those expressed in P and C. We assume that $\mathcal{R}$ is consistent. The mixed network $\mathcal{M} = (\mathcal{B},\mathcal{R})$ represents the conditional probability $P_{\mathcal{M}}(x) = P_{\mathcal{B}}(x|x \in \rho)$ if $x \in \rho$ and 0 otherwise.

Finally, capturing dynamic behavior in the model requires a further extension. Dynamic Bayesian Networks are Markov models whose state-space and transition functions are expressed in a factored form using Bayesian Networks. They are defined by a prior $P(X_0)$ and a state transition function $P(X_{t+1}|X_t)$. Hybrid Dynamic Bayesian Networks (HDBNs) allow continuous variables while Hybrid Dynamic Mixed Networks (HDMNs) also permit explicit discrete constraints.

> DEFINITION: *A* **Hybrid Dynamic Mixed Network (HDMN)** *is a pair* $(M_0, M_{\rightarrow})$, *defined over a set of variables* $X = \{x_1,...,x_n\}$, *where* M_0 *is an HMN defined over* X *representing* $P(X_0)$. $M_{\rightarrow}$ *is a 2-slice network defining the stochastic process* $P(X_{t+1}|X_t)$. *The* 2-time-slice Hybrid Mixed network (2-THMN) *is an HMN defined over* $X' \cup X''$ *such that* X' *and* X'' *are identical to* X. *The acyclic graph of the probabilistic portion is restricted so that nodes in* X' *are root nodes and have no CPDs associated with them. The constraints are defined the usual way. The 2-THMN represents a conditional distribution* $P(X''|X')$.

The semantics of any dynamic network can be understood by unrolling the network to T time-slices. Namely, $P(X_{0:t}) = P(X_0) * \prod_{t=1}^{T} P(X_t|X_{t-1})$ where each probabilistic component can be factored in the usual way, yielding a regular HMN over T copies of the state variables.

The most common tasks over Dynamic Probabilistic Networks is filtering and prediction. Filtering is the task of determining the belief state $P(X_t|e_{0:t})$ where X_t is the set of variables at time t and $e_{0:t}$ are the observations accumulated at time-slices 0 to t. Filtering can be accomplished in principle by unrolling the dynamic model and using any state-of-the art exact or approximate reasoning algorithm. The join-tree-clustering algorithm is the most commonly used algorithm for *exact* inference in Bayesian networks. Unfortunately, the stochastic nature of Dynamic Networks restricts the applicability of join-tree clustering considerably. In the discrete case the temporal structure implies tree-width which equals to the number of state variables that are connected with the next time-slice, thus making the factored representation ineffective. Even worse, when both continuous and discrete variables are present (as they are in the transportation models we consider) the effective treewidth is $O(T)$ where T is the number of time slices, thus making exact inference infeasible. The next sections discuss the extension of approximate inference algorithms for Hybrid Dynamic Mixed Networks. Section extends the Expectation Propagation algorithm, and section extends the sampling-based algorithm known as Rao-Blackwellised Particle Filtering.

EXPECTATION PROPAGATION

Expectation Propagation (EP) is an approximate inference algorithm for HDBNs. The idea in forward pass EP is to perform belief propagation by passing messages between slices t and $t+1$ along the ordering $t = 0$ to T. Noting that EP is an extension of Generalized Belief Propagation (GBP) for use with HDBNs *(13)*, we have developed a similar extension for use with HDMNs. The extension is derived by integrating the results in *(18, 7, 15, 14)*. The derivation can be summarized as modification of Iterative Join Graph Propagation (IJGP) *(7)*, a GBP algorithm that can handle mixed networks. The resulting technique, which we have termed IJGP(i)-S (where ”S” denotes that the process is sequential), permits message passing between model time slices as with EP.

IJGP performs non-sequential message passing on a join-graph, which is a collection of cliques or clusters such that the interaction between the clusters is captured by a graph. Each join-graph contains a subset of variables from the graphical model. IJGP(i) is a parameterized version of IJGP that operates on a join-graph that has less than $i+1$ discrete variables in each clique. The complexity of IJGP(i) is bounded exponentially by i, also called the i-bound. In the message-passing step of IJGP(i), a message is sent between all pairs of nodes that are neighbors of each other in the join-graph. A message sent by node N_i to N_j is constructed by multiplying all the functions and messages in a node (except the message received from N_j) and marginalizing on the common variables between N_j and N_i *(see 7)*.

IJGP(i) is easily adapted for use with HDMNs, but because we are performing online inference, we cannot recompute the join-graph every time new evidence arrives as would be done in off-line application. To handle this, we combine Murphy's *(18)* online method to compute a join-tree with Dechter et al.'s *(7)* method to compute join-graphs from join-trees. The result is an online procedure for constructing a join-graph in which the interface is split into smaller cliques such that the new cliques have less than $i+1$ variables. This construction procedure is shown in Figure 1.

[FIGURE 1 about here.]

To extend IJGP(i) to sequential applications, the algorithm is modified to handle message-passing between time slices. This modification yields IJGP(i)-S, which performs message passing sequentially as follows. At each time-slice t, we perform message-passing over nodes in t and the interface of t with $t-1$ and $t+1$ (shown by the ovals in Figure 1). The new functions computed in the interface between t and $t+1$ are then used by $t+1$, when we perform message passing in $t+1$.

Further technical details are beyond the scope of this paper, and are covered in a related paper *(12)*.

RAO-BLACKWELLISED PARTICLE FILTERING

This section extends the Rao-Blackwellised Particle Filtering algorithm *(10)* from HDBNs to HDMNs. Some details on its derivation are given below.

Particle filtering is conceptually easy to understand. Imagine a cloud of thousands of particles, each representing a single random draw from the jointly distributed unknown variables. With each time step, each particle is assigned a new state based on its prior conditions, and on the current time step's estimated probability distributions. As the simulation progresses, some particles become exceedingly unlikely while others become more and more likely.

Formally, particle filtering uses a weighted set of samples or particles to approximate the filtering distribution. Given a set of particles $X_t^1,\ldots,X_t^N$ approximately distributed according to the target-filtering distribution $P(X_t = M|e_{0:t})$, the filtering distribution is given by $P(X_t = M|e_{0:t}) = 1/N\sum_{i=1}^{N}\delta(X_t^i = M)$ where δ is the Dirac-delta function. Since we cannot sample from $P(X_t = M|e_{0:t})$ directly, particle filtering samples from an appropriate (importance) proposal distribution $Q(X)$. The particle filter starts by generating N particles according to an initial proposal distribution

$Q(X_0|e_0)$. At each step, it generates the next state X^i_{t+1} for each particle X^i_t by sampling from $Q(X_{t+1}|X^i_t, e_{0:t})$. It then computes the weight of each particle based given by $w_t = P(X)/Q(X)$ to compute a weighted distribution and then *re-samples* from the weighted distribution to obtain a set of un-biased or un-weighted particles.

Particle filtering often shows poor performance in high-dimensional spaces, as each dimension must be met by a larger set of particles. Its performance can be improved by sampling from a sub-space by using the *Rao-Blackwellisation (RB) theorem* (and then the particle filtering is called Rao-Blackwellised Particle Filtering (RBPF)) *(10, 12, 5)*.

The main difficulty with extending RBPF to HDMNs lies in correctly handling the constraints of the mixed network. The straightforward approaches result in either an unacceptably high number of particles which violate model constraints, or in computationally expensive inference procedures that try to incorporate constraints into the RBPF algorithm. Our approach is to aim for the middle ground between the two extremes. The idea is to combine the constraint network and the Bayesian Network into a single approximate distribution Ω_{app} using IJGP(i), which is a bounded inference procedure. Note that IJGP(i) we use here is different from the algorithm IJGP(i)-S described in the previous section. IJGP(i) can be derived using the results in *(7)*, *(15)*, and *(14)*. For lack of space we do not describe the details of this algorithm *(see a companion paper 11, for details)*.

Because IJGP(i) has polynomial time complexity for constant *i*, it will not eliminate all particle-rejections due to constraint violations. However, using IJGP(i) reduces the chance of sampling a non-solution, because IJGP(i) removes many inconsistent tuples *(6)*. Another important advantage of using IJGP(i) is that it yields very good approximations to the true posterior *(see a companion paper 11, for empirical results)* thereby proving to be an ideal candidate for proposal distribution.

[FIGURE 2 about here.]

The integration of the ideas described above into a formal algorithm called IJGP-RBPF is given in figure 2. It uses the same template as in *(10)* and the only step different in IJGP-RBPF from the original template is the implementation of the Sequential Importance Sampling step (SIS). More details on the SIS step implementation are given in *(12)*.

THE TRANSPORTATION MODEL

In this section, we describe the application of HDMNs to the problem of inferring travel and activity behavior of individuals. The major query in our HDMN model is to determine the goal of a traveler and the route to that goal, given the current location of the traveler's car. Similar work was described in *(17)* in a different context for detecting abnormal behavior in Alzheimer's patients; they use a Abstract Hierarchical Markov Models (AHMM) for reasoning about this problem. Our approach provides a more general modeling framework and approximate inference algorithms, as well as a domain independent implementation which allows domain expert to add and test variants of the model.

[FIGURE 3 about here.]

Figure 3 shows a HDMN model for modeling the travel activity of individuals. Note that the directed links express the probabilistic relationships while the undirected (bold) edges express the constraints. The variables in the model are as follows.

The roads are modeled as a graph $G(V,E)$, where the vertices V correspond to intersections, and the edges E correspond to segments of roads between intersections. Multiple segements occur between intersections if the road in question has one or more intermediate shape points. The variables d_t and w_t represent the information about time-of-day and day-of-week respectively. d_t is a discrete variable and has four values $(morning, afternoon, evening, night)$ while the variable w_t has two values $(weekend, weekday)$. Variable g_t represents the person's next goal. Ideally this would represent an activity. However, it is hard to imagine collecting activity information without directly questioning the traveler. Since the goal of this research is an inference model which requires little or no interaction with the driver, the goal is taken to represent a destination in space, typically a small spatial area such as a parking lot.

In the travel network graph, each goal is represented as a set of edges $E_1 \subset E$. The route variable r_t represents the route being taken by the person at time t to move from one goal to another. The number of values it can take has been arbitrarily set to $|g_t|^2$—i.e., the model can identify $|g_t|^2$ distinct routes that the individual uses to navigate the network. The person's location l_t and velocity v_t are estimated from the current GPS reading y_t. The f_t variable is a counter (essentially goal duration) that governs goal switching. The current location l_t is represented in the form of a two-tuple (a, ω). Here a is an edge in the travel network $G(V,E)$, $a = (s_1, s_2)$, $a \in E$ and $s_1, s_2 \in V$. The variable ω is a Gaussian whose mean is equal to the distance between the person's current position on a and the starting point of the segment, s_1.

To model switching between goals—the point at which a person finishes one activity and starts traveling to the next—we use a set of constraints, represented by the undirected bold lines in figure 3. We assume that a person switches his or her goal from one time slice to another when he or she is near a goal or moving away from a goal but not when he or she is on a goal location. In this latter case, goal switching is forced when a specified maximum time at that goal, or duration D, is reached. The assumptions are modeled using the following constraints. as follows:

1. If $l_{t-1} = g_{t-1}$ and $f_{t-1} = 0$ Then $f_t = D$,
2. If $l_{t-1} = g_{t-1}$ and $f_{t-1} > 0$ Then $f_t = f_{t-1} - 1$,
3. If $l_{t-1} \neq g_{t-1}$ and $f_{t-1} = 0$ Then $f_t = 0$,
4. If $l_{t-1} \neq g_{t-1}$ and $f_{t-1} > 0$ Then $f_t = 0$,
5. If $f_{t-1} > 0$ and $f_t = 0$ Then g_t is given by $P(g_t|g_{t-1})$,
6. If $f_{t-1} = 0$and $f_t = 0$ Then g_t is same as g_{t-1},
7. If $f_{t-1} > 0$ and $f_t > 0$ g_t is same as g_{t-1}, and
8. If $f_{t-1} = 0$ and $f_t > 0$ g_t is given by $P(g_t|g_{t-1})$.

EXPERIMENTAL RESULTS

The test data consists of a log of GPS readings collected by one of the authors. The GPS readings represent the normal daily travel of two different vehicles. The test data was collected over a six month period at intervals of 1-5 seconds each. The data consist of the current time, date, location and velocity of the person's travel. The location is given as latitude and longitude pairs.

We used the first three months' data as our training set while the remaining data was used as a test set. TIGER/Line files available from the US Census Bureau formed the graph on which the data was snapped. The data was first divided into individual routes taken by each person, and the HDMN model was learned using the Monte Carlo version of the EM algorithm *(17, 16)*. Goals were approximated by identifying all destinations (GPS coordinates) at which the dwell time was greater than 15 minutes. These points were then clustered to form contiguous destinations. As specified earlier the aim of the model is two-fold: (a) Finding the destination or goal of a person given the current location and (b) Finding the route taken by the person towards the destination or goal.

To compare our inference and learning algorithms, we use three HDMN models. Model-1 is the model shown in Figure 3. Model-2 is the model given in Figure 3 with the variables w_t and d_t removed from each time-slice. Model-3 is the base-model which tracks the person without any high-level information and is constructed from Figure 3 by removing the variables w_t, d_t, f_t, g_t and r_t from each time-slice. In other words, Model-3 essentially just snaps the GPS point y_t to a network location l_t at some speed v_t, and then tries to predict the sequence of links that the traveler will take.

We used 4 inference algorithms. Since EM-learning uses inference as a sub-step, we have 4 different learning algorithms. We call these algorithms as IJGP-S(1), IJGP-S(2) and IJGP-RBPF(1,1,N) and IJGP-RBPF(1,2,N) respectively. Note that the algorithm IJGP-S(i) (described in section) uses i as the i-bound. IJGP-RBPF(i,w,N) (described in section) uses i as the i-bound for IJGP(i), w as the w-cutset bound and N is the number of particles at each time slice. Note that for Model-1, we only use IJGP-RBPF(1,1) and IJGP(1)-S because the maximum i-bound in this model is bounded by 1 (see section). Three values of N were used: 100, 200 and 500. For EM-learning, N was 500. Experiments were run on a Pentium-4 2.4 GHz machine with 2G of RAM. While the ultimate goal of this research is to embed the process in a PDA or in-vehicle computer, at this early stage of development it is unrealistic to expect the algorithms to be optimized enough to run on anything but a high-powered work station.

Learning was performed offline. The slowest learning algorithm based on GBP-RBPF(1,2) used almost 5 days of CPU time for Model-1, and almost 4 days for Model-2. While this is far from instantaneous, it is significantly less than the period over which the data was collected. This is promising in that it is reasonable to expect that in practice an inference model can be updated each night with the results of the day's travel.

Finding destination or Goal of a person

The results for goal prediction with various combinations of models, learning and inference algorithms are shown in Tables 1, 2 and 3. Prediction accuracy is defined as the number of goals predicted correctly. The column "Time" in Tables 1, 2 and 3 shows the time for inference algorithms in seconds, while the other entries indicate the accuracy for each combination of inference and learning algorithms.

In terms of which model yields the best accuracy, we can see that Model-1 achieves the highest prediction accuracy of 84% while Model-2 and Model-3 achieve prediction accuracies of 77% and 68% respectively or lower. The careful reader may wonder how goals are predicted using Model-3, when the goal variable g is not included in that model. The answer is that the model predicts a sequence of segments which end at a destination. While the hidden variable g is not explicitly included in the model, the output of the model still gets to the correct destination two-thirds of the time. Adding the hidden variables for goals, goal-switching, and routes to the model (Model-2) improves the accuracy to three-quarters of the time, and adding time of day and day of week boost the accuracy yet again to just over 80%. These results are heartening in that the activity model relies only upon what is observable *without* direct user interaction. Further, other, known data sources such as land use statistics were not included in the model, and the goal variable was strictly spatial rather than activity-based—all factors that might have led to poor model performance.

Recall that the modeling process has two steps. In the learning step the various parameters are trained using known data. Then in the inference step, predictions are made for the three months of data not used in the training set. For Model-1, to verify which algorithm yields the best learned model we see that IJGP-RBPF(1,2) and IJGP(2)-S yield an accuracy of 83% and 81% respectively while for Model-2, we see that the average accuracy of IJGP-RBPF(1,2) and IJGP(2)-S was 76% and 75% respectively. From these two results, we can see that IJGP-RBPF(1,2) and IJGP(2)-S both yield more or less equivalent results for the learning step. That is, they produce models which perform equivalently in the inference stage.

For Model-1 and Model-2, to verify which algorithm yields the best inference accuracy given a learned model, we see that IJGP(2)-S is the most cost-effective alternative in terms of time versus accuracy while IJGP-RBPF yields the best accuracy. Note that accuracy and runtime with the particle-filtering algorithm IJGP-RBPF both increase with increasing numbers of paricles.

[TABLE 1 about here.]

[TABLE 2 about here.]

[TABLE 3 about here.]

[TABLE 4 about here.]

Finding the Route taken by the person

To see how our models predict a person's route, we use the following method. We first run our inference algorithm on the learned model and predict the route that the person is most likely to take. We then superimpose this route on the actual route taken by the person. We then count the number of roads that were not taken by the person but were in the predicted route, i.e., the false positives, and divide that by the total number of links in the predicted route to get the FP fraction. The number of roads that *were* taken by the person but were *not* in the predicted route, the false negatives, are then tallied up for all of the non-training data and divided by the total number of links in the actual routes to arrive at the FN fraction. Both of these figures are calculated over the entire three months of non-training data. Because these are averages, the goals predicted incorrectly have correspondingly more drastic route errors, which tends to skew the results towards large FP and FN. When the goals are predicted correctly, the routes are typically much closer to what is observed.

The two measures (FP and FN) are reported in Table 4 for the best performing learning models in each category: viz GBP-RBPF(1,2) for Model-1 and Model-2 and GBP-RBPF(1,1) for Model-3. In this table, the number 31 false positives (FP) means 31% of the links inferred for the non-training data were inferred incorrectly. As we can see Model-1 and Model-2 have the best route prediction accuracy (given by low false positives and false negatives). Again, given the importance of predicting the goal accurately in decreasing the number of route-level errors, it stands to reason that these models would outperform Model-3, which has no notion of goals or routes embedded in it. Finally, note that in practice an in-vehicle device that inferred a route different from what was being observed would probably stop trying to predict where the driver was going, and instead go into some “new behavior” or “route alteration” type of mode. When the predictions are correct and continue to match up to what is observed, the box can concentrate on tasks such as observing the expected links ahead for incidents, and/or alerting the parties at the predicted destination with an expected arrival time.

RELATED WORK

(17) and *(19)* describe a model based on AHMEM and Hierarchical Markov Models (HMMs) respectively for inferring high-level behavior from GPS-data. Our model goes beyond their model by representing two new variables day-of-week and time-of-day which improves the accuracy in our model by about 6%.

A mixed network framework for representing deterministic and uncertain information was presented in *(8)*, *(14)*, and *(9)*. These previous works also describe exact inference algorithms for mixed networks with the restriction that all variables should be discrete. This research goes beyond these previous works in that we describe approximate inference algorithms for the mixed network framework, allow continuous Gaussian nodes with certain restrictions in the mixed network framework and model discrete-time stochastic processes. The approximate inference algorithms called IJGP(i) described in *(7)* handled only discrete variables. Here this algorithm is extended to include Gaussian variables and discrete constraints. We also develop a sequential version of this algorithm for dynamic models.

Particle Filtering is a very attractive research area *(10)*. Particle Filtering in HDMNs can be inefficient if non-solutions of the constraint portion of the model have a high probability of being sampled. We show how to alleviate this difficulty by performing IJGP(i) before sampling. This algorithm IJGP-RBPF yields the best performance in our settings and might prove to be useful in applications in which particle filtering is preferred.

CONCLUSION AND FUTURE WORK

In this paper, we introduced a new modeling framework called HDMNs, a representation that handles discrete-time-stochastic processes, deterministic and probabilistic information on both continuous and discrete variables in a systematic way. We also propose a GBP-based algorithm called IJGP(i)-S for approximate inference in this framework, and present a class of Rao-Blackwellised particle filtering algorithm, IJGP-RBPF, for effective sampling of HDMNs in the presence of constraints.

The framework and two approaches to learning and inference are then applied to modeling travel behavior. This domain is attractive because travel behavior is highly routinized, but at the same time is highly variable. People may make daily trips between home and work, but those trips are varied in time and space, and interspersed with other trips. The eventual goal of this research is to be able to infer exactly the routine portions of travel behavior, and also to be able to infer exactly when routines have been broken. This goal has applicability to designing travel behavior surveys, as well as to desinging in-vehicle or hand-held travel assistance devices.

ACKNOWLEDGEMENTS

The first and third authors were supported in part by National Science Foundation under award numbers 0331707 and 0331690. The second author was supported in part by the NSF grant IIS-0412854. The fourth and fifth authors were supported in part by Caltrans as part of the California Advanced Transportation Management Systems Testbed.

REFERENCES

[1] (1995). The household activity pattern problem: General formulation and solution. *Transportation Research, Part B: Methodological*, 29B(1):61–77.

[2] (2000). *ALBATROSS: a learning based transportation oriented simulation system*. European Institute of Retailing and Services Studies, Eindhoven.

[3] (2001). A bridge between travel demand modeling and activity-based travel analysis. *Transportation Research, Part B: Methodological*, 35B(5):481–506.

[4] Arentze, T. and Timmermans, H. (2003). Modeling learning and adaptation processes in activity travel choice: A framework and numerical experiments. *Transportation*, 30:37–62.

[5] Bidyuk, B. and Dechter, R. (2004). On fi nding minimal w-cutset problem. In *UAI-04*.

[6] Dechter, R. (2003). *Constraint Processing*. Morgan Kaufmann.

[7] Dechter, R., Kask, K., and Mateescu, R. (2002). Iterative join graph propagation. In *UAI '02*, pages 128–136. Morgan Kaufmann.

[8] Dechter, R. and Larkin, D. (2001). Hybrid processing of beliefs and constraints. In *Proc. Uncertainty in Artifi cial Intelligence*, pages 112–119.

[9] Dechter, R. and Mateescu, R. (2004). Mixtures of deterministic-probabilistic networks and their and/or search space. In *Proceedings of the 20th Annual Conference on Uncertainty in Artifi cial Intelligence (UAI-04)*.

[10] Doucet, A., de Freitas, N., Murphy, K. P., and Russell, S. J. (2000). Rao-blackwellised particle fi ltering for dynamic bayesian networks. In *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artifi cial Intelligence*, pages 176–183. Morgan Kaufmann Publishers Inc.

[11] Gogate, V. and Dechter, R. (2005). Approximate inference algorithms for hybrid mixed networks. Technical report, Donald Bren School Of Information and Computer Science, University of California, Irvine.

[12] Gogate, V., Dechter, R., Bidyuk, B., Rindt, C., and Marca, J. (2005). Modeling transportation routines using hybrid dynamic mixed networks. In *Proceedings of the 21st Annual Conference on Uncertainty in Artifi cial Intelligence (UAI-05)*. forthcoming.

[13] Heskes, T. and Zoeter, O. (2002). Expectation propagation for approximate inference in dynamic bayesian networks.

[14] Larkin, D. and Dechter, R. (2003). Bayesian inference in the presence of determinism. In *AI-STAT*.

[15] Lauritzen, S. (1992). Propagation of probabilities, means, and variances in mixed graphical association models. *Journal of the American Statistical Association*, 87(420):1098–1108.

[16] Levine, R. and Casella, G. (2001). Implementations of the monte carlo em algorithm. *Journal of Computational and Graphical Statistics*, 10:422–439.

[17] Liao, L., Fox, D., and Kautz., H. (2004). Learning and inferring transportation routines. In *AAAI-2004*.

[18] Murphy, K. (2002). *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California Berkeley.

[19] Patterson, D. J., Liao, L., Fox, D., and Kautz, H. (2003). Inferring high-level behavior from low-level sensors. In *Proceedings of UBICOMP 2003: The Fifth International Conference on Ubiquitous Computing*, pages 73–89.

[20] Srinivasan, K. K. and Mahmassani, H. S. (2002). Dynamic decision and adjustment processes in commuter behavior under real-time information. Technical report, Center for Transportation Research, UT Austin.

[21] Srinivasan, K. K. and Mahmassani, H. S. (2005). A dynamic kernel logit model for the analysis of longitudinal discrete choice data: Properties and computational assessment. *Transportation Science*, 39(2):160–181.

LIST OF TABLES

1 Goal-prediction: Model-1 15
2 Goal Prediction: Model-2 16
3 Goal Prediction Model-3 17
4 False positives (FP) and False negatives for routes taken by a person (FN) 18

TABLE 1: Goal-prediction: Model-1

			LEARNING			
			IJGP-RBPF		IJGP-S	
N	Inference	Time	(1,1)	(1,2)	(1)	(2)
100	IJGP-RBPF(1,1)	12.3	78	80	79	80
100	IJGP-RBPF(1,2)	15.8	81	84	78	81
200	IJGP-RBPF(1,1)	33.2	80	84	77	82
200	IJGP-RBPF(1,2)	60.3	80	84	76	82
500	IJGP-RBPF(1,1)	123.4	81	84	80	82
500	IJGP-RBPF(1,2)	200.12	84	84	81	82
	IJGP(1)-S	9	79	79	77	79
	IJGP(2)-S	34.3	74	84	78	82
	Average		79.625	82.875	78.25	81.25

TABLE 2: Goal Prediction: Model-2

			LEARNING			
			IJGP-RBPF		IJGP-S	
N	Inference	Time	(1,1)	(1,2)	(1)	(2)
100	IJGP-RBPF(1,1)	8.3	73	73	71	73
100	IJGP-RBPF(1,2)	14.5	76	76	71	75
200	IJGP-RBPF(1,1)	23.4	76	77	71	75
200	IJGP-RBPF(1,2)	31.4	76	77	71	76
500	IJGP-RBPF(1,1)	40.08	76	77	71	76
500	IJGP-RBPF(1,2)	51.87	76	77	71	76
	IJGP(1)-S	6.34	71	73	71	74
	IJGP(2)-S	10.78	76	76	72	76
	Average		75	75.75	71.125	75.125

TABLE 3: Goal Prediction Model-3

			LEARNING	
N	Inference	Time	IJGP-RBPF(1,1)	IJGP(1)-S
100	IJGP-RBPF(1,1)	2.2	68	61
200	IJGP-RBPF(1,1)	4.7	67	64
500	IJGP-RBPF(1,1)	12.45	68	63
	IJGP(1)-S	1.23	66	62
	Average		67.25	62.5

TABLE 4: False positives (FP) and False negatives for routes taken by a person (FN)

		Model1	Model2	Model3
N	INFERENCE	FP/FN	FP/FN	FP/FN
	IJGP(1)	33/23	39/34	60/55
	IJGP(2)	31/17	39/33	
100	IJGP-RBPF(1,1)	33/21	39/33	60/54
200	IJGP-RBPF(1,1)	33/21	39/33	58/43
100	IJGP-RBPF(1,2)	32/22	42/33	
200	IJGP-RBPF(1,2)	31/22	38/33	

LIST OF FIGURES

1 Schematic illustration of the Procedure used for creating join-graphs and join-trees of HDMNs . . . 20
2 IJGP-RBPF for HDMNs . . . 21
3 Car Travel Activity model of an individual . . . 22

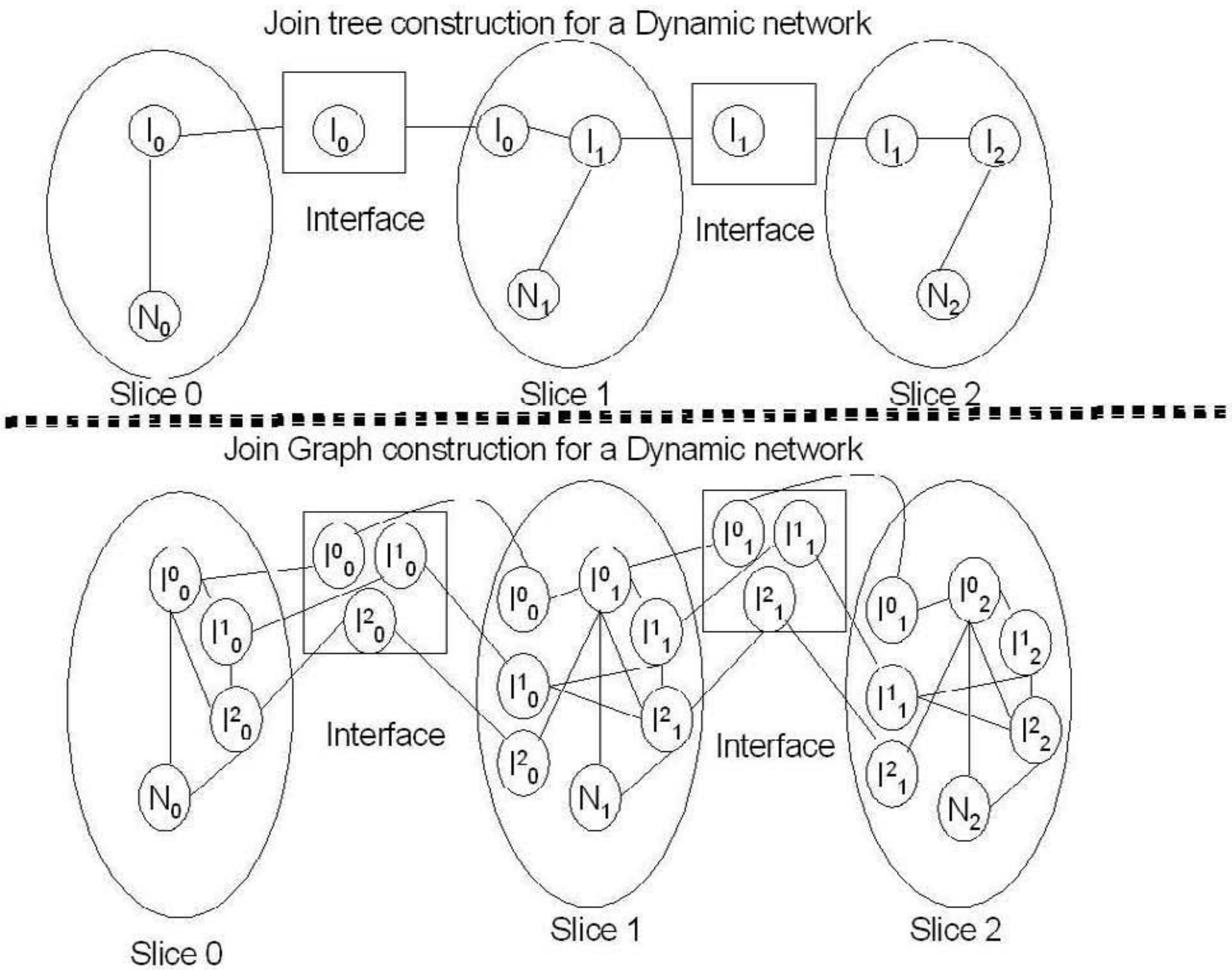

FIGURE 1: Schematic illustration of the Procedure used for creating join-graphs and join-trees of HDMNs

Algorithm IJGP-RBPF

- **Input:** A Hybrid Dynamic Mixed Network $(X,D,G,P,C)_{0:T}$ and an observation sequence $e_{0:T}$ Integer N, w and i.
- **Output:** $P(X_T|e_{0:T})$
- For t = 0 to T do
- **Sequential Importance Sampling step**:
  1. **Generalized Belief Propagation step**
     Use IJGP(i) to compute the proposal distribution Ω_{app}
  2. **Rao-Blackwellisation step**
     Partition the Variables X_t into R_t and Z_t such that the treewidth of a join-tree of Z_t is w.
  3. **Sampling step**
     For i = 1 to N do
     (a) Generate a R_t^i from Ω_{app}.
     (b) reject sample if r_t^i is not a solution. i=i-1;
     (c) Compute the importance weights w_t^i of R_t^i.
  4. Normalize the importance weights to form $\widehat{w_t^i}$.
- **Selection step**:
  - Resample N samples from $\widehat{R_t^i}$ according to the normalized importance weights $\widehat{w_t^i}$ to obtain new N random samples.
- **Exact step**:
  - for i = 1 to N do
    Use join-tree-clustering to compute the distribution on Z_t^i given Z_{t-1}^i, e_t, $\widehat{R_t^i}$ and $\widehat{R_{t-1}^i}$.

FIGURE 2: IJGP-RBPF for HDMNs

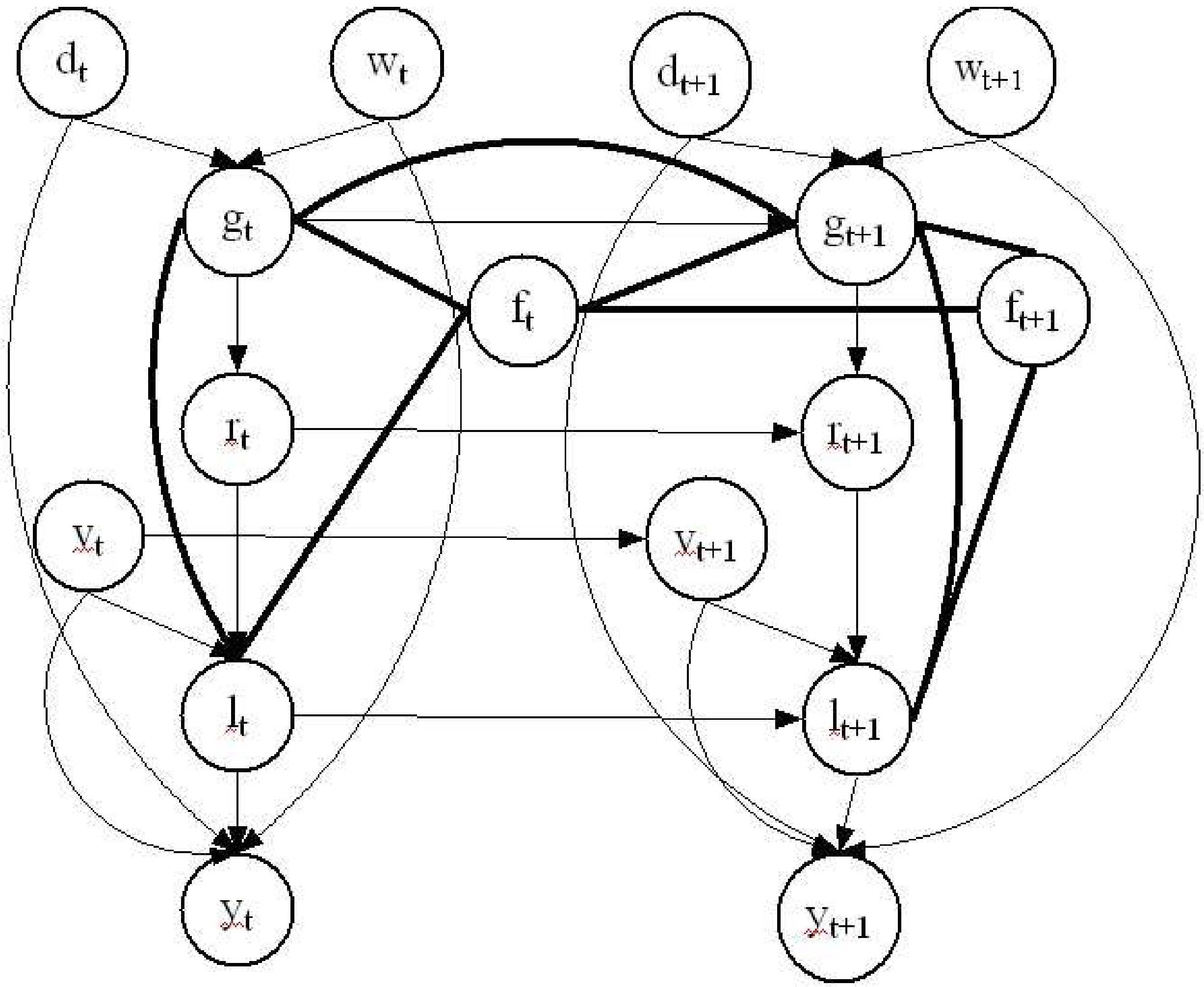

FIGURE 3: Car Travel Activity model of an individual