

A Logic and Time Nets for Probabilistic Inference

Keiji Kanazawa*

Department of Computer Science
Brown University, Box 1910
Providence, RI 02912
kgk@cs.brown.edu

Abstract

In this paper, we show a new approach for reasoning about time and probability that combines a formal declarative language with a graph representation of systems of random variables for making inferences. First, we provide a continuous-time logic for expressing knowledge about time and probability. Then, we introduce the *time net*, a kind of Bayesian network for supporting inference with statements in the logic. Time nets encode the probability of facts and events over time. We provide a simulation algorithm to compute probabilities for answering queries about a time net. Finally, we consider an incremental *probabilistic temporal database* based on the logic and time nets to support temporal reasoning and planning applications. The result is an approach that is semantically well-founded, expressive, and practical.

Introduction

We are interested in the design of robust inference systems for supporting activity in dynamic domains. In most domains, things cannot always be predicted accurately in advance. Thus, the capability to reason about change in an uncertain environment remains an important component of robust performance. Our goal is to develop a computational theory for temporal reasoning under uncertainty that is well suited to a wide variety of domains.

To this end, in past work [Dean and Kanazawa, 1988; Dean and Kanazawa, 1989a], we initiated inquiry into the application of probability theory to temporal reasoning and planning. We chose probability theory, because it is the most well understood calculus for uncertainty [Shafer and Pearl, 1990]. This effort to develop

*This work was supported in part by a National Science Foundation Presidential Young Investigator Award IRI-8957601 with matching funds from IBM, by the Advanced Research Projects Agency of the Department of Defense monitored by the Air Force Office of Scientific Research under Contract No. F49620-88-C-0132, and by the National Science foundation in conjunction with the Advanced Research Projects Agency of the Department of Defense under Contract No. IRI-8905436.

probabilistic temporal reasoning came alongside complementary developments by other researchers [Hanks, 1990; Weber, 1989; Berzuni *et al.*, 1989].

In probabilistic temporal reasoning, we have been interested first and foremost in projection tasks. As an example, consider a scenario taken after [Dean and Kanazawa, 1989b]. We are at a warehouse. Sally the truck driver arrives at 2pm, expecting to pick up cargo en route to another warehouse. Trucks arrive to pick up and deliver cargo all day long; a long time can pass before a truck is taken care of. What is the chance that Sally will become fed up and leave without picking up her cargo? What is the chance of her staying in the loading dock 10 minutes? 30 minutes? Should we load Sally's truck earlier than than Harry's, who arrived 10 minutes earlier?

In this paper, we extend our past work by defining new languages for expressing knowledge about time and probability, and by designing an algorithm for answering queries about the probability of facts and events over time. First, we present a temporally-quantified propositional logic of time and probability. Then we present the time net, a graph representation of a system of random variables developed for inference about knowledge expressed in the logic. We conclude by considering an incremental probabilistic temporal database based on the logic and time nets.

A Logic of Time and Probability

In this section, we describe a logic for expressing statements about probability and time. Our primary goal is to design a language to support expression of facts and questions such as arising in the warehouse scenario. The key entities of interest are the probability of facts and events over time.

Our logic of time and probability combines elements of the logics of time by Shoham [Shoham, 1988] with elements of the logics of probability by Bacchus [Bacchus, 1988] and Halpern [Halpern, 1989]. Basically, the features related to time are borrowed from Shoham, and the features related to probability are borrowed from Bacchus and Halpern. The resulting logic, called \mathcal{L}_{cp} , is similar in many respects to the logic of time

and chance by Haddawy [Haddawy, 1990].

\mathcal{L}_{cp} is a continuous-time logic for reasoning about the probability of facts and events over time. It is temporally-quantified, and has numerical functions for expressing probability distributions. \mathcal{L}_{cp} is otherwise propositional, as far as the basic facts and events that it addresses. In this paper, we concentrate on the basic ontology and syntax of \mathcal{L}_{cp} . A detailed exposition of a family of logics of time and probability, both propositional and first order, complete with their semantics and a discussion of complexity, is given in [Kanazawa, Forthcoming].

In \mathcal{L}_{cp} , there is a set of propositions that represent facts or events, such as `here(Sally)` and `arrive(Sally)`, intended to represent, respectively, the fact of Sally being here, and the event of Sally arriving. A proposition such as `here(Sally)` taken alone is a fact (or event) *type*, the generic condition of Sally being here. Each particular instance of `here(Sally)` must be associated with time points, *e.g.*, via the `holds` sentence former, in a statement such as `holds(2pm, 3pm, here(Sally))`. Such an instance is a fact (or event) *token*. Each token refers to one instance of its type. Thus if Sally comes to the warehouse twice on Tuesday, there will be two tokens of type `here(Sally)`, which might be referred to as `here1(Sally)` and `here2(Sally)`. Where unambiguous, we commonly use the same symbol to refer to both a token and its type.

Facts and events have different status in \mathcal{L}_{cp} . A fact is something that once it becomes true, tends to stay true for some time. It is basically a *fluent* [McCarthy and Hayes, 1969]. A fact is true for some interval of time $[t_0, t_1]$ if and only if it is true for all subintervals $[u, v]$ where $t_0 \leq u < v \leq t_1$.

By contrast, an event takes place instantaneously; it stays true only for an infinitesimally small interval of time. We restrict events to be one of three kinds: *persistence causation* [McDermott, 1982] or `enab`, *persistence termination* or `clip`, or *point event*. The first two are, respectively, the event of a fact becoming true, and the event of a fact becoming false. For the fact type `here(Sally)`, the `enab` event type is written `beg(here(Sally))` and the `clip` event type is written `end(here(Sally))`.

A point event is a fact that stays true for an instant only, such as Sally's arrival. In \mathcal{L}_{cp} 's ontology, a point event at time t is a fact that holds true for an infinitesimally small interval of time $[t, t+\epsilon]$, $\epsilon > 0$. Although a point event such as `arrive(Sally)` is really a fact, because it is only true for an instant, we usually omit its `enab` and `clip` events.

In \mathcal{L}_{cp} sentences, a fact or event token is associated with an interval or instant of time. Facts are associated with time by the `holds` sentence former as we saw earlier, *e.g.*, `holds(2pm, 3pm, here(Sally))`, and events are associated with a time instant with the `occ` sentence former, *e.g.*, `occ(t, arrive(Sally))` or

`occ(t, beg(here(Sally)))`. `occ(t, ϵ)` is equivalent to `holds(t, t+ ϵ , ϵ)` for some $\epsilon > 0$.

Sentences about probability are formed with the `P` operator. For instance, the probability that Sally arrives at 2pm is $P(\text{occ}(2\text{pm}, \text{arrive}(\text{Sally})))$. The probability that she never arrives is written as $P(\text{occ}(+\infty, \text{arrive}(\text{Sally})))$. The probability that Sally is here between 1pm and 2pm is $P(\text{holds}(1:00, 2:00, \text{here}(\text{Sally})))$.

With continuous time, the probability of a random variable at a point in time is always 0. Because of our definition of events as facts that hold for ϵ , this is not problematic. An alternative is to introduce probability density functions into the language, and define probability in terms of such functions [Kanazawa, Forthcoming]. The probability that a fact holds for an arbitrary interval, $P(\text{holds}(u, v, \varphi))$, is given by

$$P(\text{occ}(b, \text{beg}(\varphi)) \wedge \text{occ}(e, \text{end}(\varphi)) \\ \wedge b \leq u \wedge v \leq e)$$

Knowledge about how events cause other events, and how events follow other events, are expressed by conditional probability statements. Let φ stand for the proposition `here(Sally)`. Then, the probability that Sally is still here, given that she arrived some time ago, can be written as $P(\text{holds}(\cdot, \text{now}, \varphi) \mid \text{occ}(\cdot, \text{beg}(\varphi)))$. A function that describes such probability of a fluent's persistence is often called a *survivor function* [Cox and Oakes, 1984].

In the semantics for \mathcal{L}_{cp} , the meaning of a token is the time over which it holds or at which it occurs. In other words, the meaning of a fact token `here(Sally)` is the interval over which it holds, and the meaning of an event token `arrive(Sally)` is the instant at which it takes place. The functions `range` and `date` are used to denote those times, *i.e.*, `range(here(Sally))` and `date(arrive(Sally))`.

More precisely, \mathcal{L}_{cp} has possible worlds semantics, and the meaning of a token is a set of interval - world pairs $\{([t_{00}, t_{01}], w_0), ([t_{10}, t_{11}], w_1), \dots\}$, where the interval is the times over which the token holds or occurs in that world (actually, a potential world *history*, or *chronicle* [McDermott, 1982]). This interpretation is similar to that in [McDermott, 1982] and [Shoham, 1988]. Together with a *probability measure* over the chronicles, this defines the semantics for statements about probability in \mathcal{L}_{cp} . For temporal logics that have similar semantics, see [Haddawy, 1990].

Syntax of \mathcal{L}_{cp}

\mathcal{L}_{cp} is a three-sorted logic, with a different sort that corresponds to each of the different types of things that we wish to represent, domain objects, time, and probability. These are the object, time, and field sorts.

The object sort \mathcal{O} is used to represent objects of interest in the domain. It is a countable collection of objects, including things like Sally. Object constants and predicates are used to form propositions such as

here(Sally). The time sort \mathcal{T} is used for expressing time. \mathcal{L}_{cp} is temporally-quantified, and contains both time constants and variables, and the date function. The operator \prec defines time precedence.

The field sort \mathcal{F} is used to represent real-numbered probability. In addition to constants, the field sort contains functions for representing probability distributions. An example is $\text{nexp}(t, 0.75)$, which is the negative exponential distribution for time t with the parameter denoted by the field constant 0.75. As in this example, field functions range over both time and field sorts, for specification of probability functions of time.

We now present the syntax for our logic \mathcal{L}_{cp} . In the following, a $n(i, j)$ -ary function is a function of arity n , with i time sort arguments and j field sort arguments, where $n = i + j$, and both $i, j \geq 0$. Given this, we can provide an inductive definition of \mathcal{L}_{cp} starting with the terms.

First of all, a single object constant is an \mathcal{O} -term.

If p is an n -ary object predicate symbol and o_1, \dots, o_n are \mathcal{O} -terms, then $p(o_1, \dots, o_n)$ is a *proposition*. The set of all propositions is \mathcal{P} . If $\pi \in \mathcal{P}$, then $\text{beg}(\pi)$ and $\text{end}(\pi)$ are *events*. The set of all events is \mathcal{E} . \mathcal{E} includes the *point events*, a subset of \mathcal{P} .

A single time variable or constant is a \mathcal{T} -term. If ϵ is an event, then $\text{date}(\epsilon)$ is a \mathcal{T} -term. A single field constant is an \mathcal{F} -term. If g is an $n(i, j)$ -ary field function symbol, t_1, \dots, t_i are \mathcal{T} -terms, and t_{i+1}, \dots, t_n are \mathcal{F} -terms, then $g(t_1, \dots, t_n)$ is an \mathcal{F} -term.

The well-formed formulas (wffs) are given by:

- If t_1 and t_2 are \mathcal{T} -terms, then $t_1 \prec t_2$ is a wff.
- If t_1 and t_2 are \mathcal{T} -terms, and $\pi \in \mathcal{P}$, then $\text{holds}(t_1, t_2, \pi)$ is a wff.
- If t is a \mathcal{T} -term, and $\epsilon \in \mathcal{E}$, then $\text{occ}(t, \epsilon)$ is a wff.
- If φ is a wff not containing the P operator, and f is an \mathcal{F} -term, then $P(\varphi) = f$ is a wff.

We add standard wffs involving logical connectives and quantifiers, as well as those involving an extended set of time relations such as \succ and \in , the latter of which denotes temporal interval membership. Conditional probability is defined as $P(\varphi | \psi) \triangleq P(\varphi \wedge \psi) / P(\psi)$, where φ and ψ are wffs.

Some examples of \mathcal{L}_{cp} sentences follow. The predicate always that is always true is defined by: $\forall t_1, t_2 \text{ holds}(t_1, t_2, \text{always})$. It has the property: $\forall t_1, t_2 P(\text{holds}(t_1, t_2, \text{always})) = 1.0$. The chance that Sally arrives at a certain time today might be given by

$\forall t \in \text{today}$

$$P(\text{occ}(t, \text{arrive}(\text{Sally}))) = \text{norm}(2\text{pm}, 0:15, t, \epsilon)$$

where $\text{norm}(\mu, \sigma, t)$ is a field function giving the normal distribution $\mathcal{N}(\mu, \sigma)$ for each time t , where today is an abbreviation for the time interval that comprises the current date, and ϵ is the duration of instantaneous events. The chance that Sally leaves without her cargo,

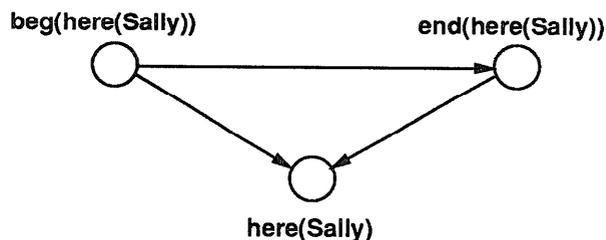


Figure 1: A time net for here(Sally).

given that she arrives at 2pm might be given by the negative exponential distribution with parameter 0.75.

$\forall t \in \text{today } P(\text{occ}(t, \text{leave/load}(\text{Sally})))$

$$\text{occ}(2\text{pm}, \text{arrive}(\text{Sally})) = \text{nexp}(2\text{pm}, t, 0.75, \epsilon)$$

Time Nets

Our goal is to construct a probabilistic temporal database for making inferences and supporting queries based on knowledge expressed in a language such as \mathcal{L}_{cp} . In this section, we show how to represent knowledge for making inferences about time and probability in graphs called *time nets*.

The time net belongs to the popular class of directed graph representation of knowledge about probability often known as the *Bayesian network* [Pearl, 1988]. A time net is a directed acyclic graph $G = (N, A)$, consisting of a set of *nodes* N , and a set of *arcs* A . We often refer to a node N_R representing a random variable R by the random variable itself. Arcs are directed edges that represent direct dependence between random variables. *Parents* and *children* of nodes, on the basis of arc direction, are defined in the usual fashion, as are *root* and *leaf* nodes.

In a time net, we allow nodes representing random variables that have values that are discrete or continuous. In addition, there are *range nodes*, which are bivariate nodes representing continuous non-negative length subsets of the real line. As an example, a range node might represent the probability of the range of body weight of an individual. The use of range variables in time nets is for representing time intervals as random variables.

Each node has a distribution for the random variable that it represents. As usual, there are marginal distributions for root nodes, and conditional distributions for other nodes. For details on the workings of Bayesian networks, we refer the reader to [Pearl, 1988].

A time net is used to represent probabilities about fact and event tokens in a body of \mathcal{L}_{cp} sentences as nodes, and direct conditional dependence between facts and events (on the basis of conditional distributions) as arcs. The random variables that a time net represents are the date of event tokens, and the range of fact tokens. For each point event token, we create a continuous valued node, a date *node*, representing the

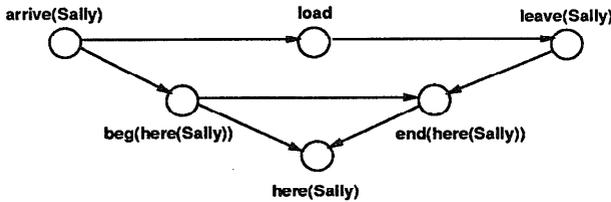


Figure 2: A simple time net for arrive and load.

time at which the event occurs. The distribution for a date node is the marginal distribution for the probability of the event. For each fact token, we create a range valued node, a *range node*, representing the interval over which the fact holds. The distribution for a range node is the marginal distribution for the interval over which the fact holds. For each fact token, we also create two date nodes representing the enab and clip events for the fact token, the *enab node* and *clip node*. A time net corresponding to the fact token *here(Sally)* is shown in figure 1. For convenience, we name a date node by its event token, and a range node by its fact token. For a given fact token π , we commonly refer to its clip node as the clip node of π 's enab node, and so on.

The time net is similar to the *network of dates* of Berzuni and Spiegelhalter [Berzuni, to appear]. The major difference is that we provide range nodes, and that we have the enhanced ontology of the enab and clip events that we use to represent knowledge embodied in sentences such as in \mathcal{L}_{cp} .

There is clear dependence between the three types of nodes for a fact token. The range node of a fact token depends on its enab and clip nodes because the range of the former is given by values of the latter. Furthermore, a clip node depends on its enab node always because a fact can only become false after it becomes true, and sometimes because we know how long the fact persists. For example, we may know that a telephone weather report always ends 60 seconds after it begins. Note that the arcs between the nodes could conceivably be reversed. For instance, if we know the duration of some fact, such as a concert, and the time at which it ended, we can infer the time at which this concert began. However, note that in general, we do not expect to have the distribution corresponding to a range node.

A simple example of a more general time net is shown in figure 2. This represents a scenario with arrive, load, and leave point events, and the fact token *here(Sally)*. The fluent *here(Sally)* is caused by *arrive(Sally)*. The latter event also causes *load* to occur, which in turn causes *leave(Sally)*. Finally, *leave(Sally)* causes *here(Sally)* to terminate. The interval over which *here(Sally)* holds is $[\text{beg}(\text{here}(\text{Sally})), \text{end}(\text{here}(\text{Sally}))]$. Typically, there are delays between, *arrive(Sally)* and

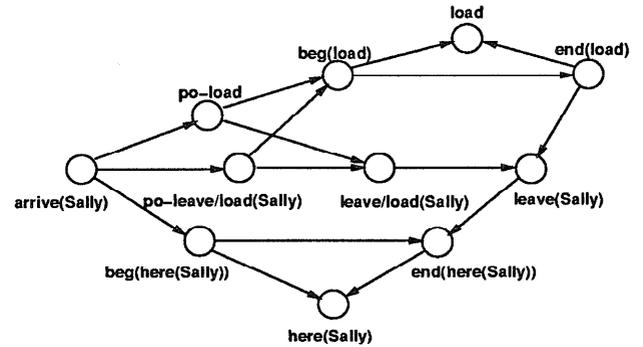


Figure 3: A time net with potential dates.

load, and between *load* and *leave(Sally)*, so that *here(Sally)* is not an instantaneous event.

A time net showing a more complex example of the same scenario is shown in figure 3. This is essentially the example given in the beginning of the paper. Unlike the previous case, where *load* was guaranteed to occur, and *leave* always followed *load*, in this case, Sally may become impatient and *leave* without her cargo. Furthermore, *load* is not an instantaneous event, but a fact that takes some time to complete.

This example potentially introduces circularities into the dependencies. For example, *beg(load)* depends on Sally not having left, and Sally leaving depends on both *beg(load)* not having taken place, and on *beg(load)* having taken place already (causing *end(load)*). To handle this, first of all, it is necessary to separate out the different cases of Sally leaving. This is done by inventing the new event type "Sally leaves without cargo". We name this event *leave/load(Sally)*. Sally actually leaves if either *end(load)* or *leave/load(Sally)* occurs. The circularity is handled with a new kind of event, the *potential date* [Berzuni, to appear]. A potential date is the date at which an event would take place, if something does not prevent it from happening. For example, *po-load*, the potential date of *beg(load)*, is the time at which *beg(load)* would take place if Sally does not become impatient and leave. Similarly, *po-leave/load(Sally)* is the time at which Sally becomes completely impatient, if *beg(load)* does not take place. The date of the actual *beg(load)* is *po-load* iff $\text{po-load} < \text{po-leave/load}(\text{Sally})$; otherwise, it is $+\infty$. Similarly, *leave* takes place at *po-leave/load(Sally)* iff $\text{po-leave/load}(\text{Sally}) < \text{po-load}$; otherwise, it takes place at some fixed time after *end(load)*.

Simulation Algorithm

In this section, we show a simple algorithm for computing densities in a time net. A time net represents knowledge about time, probability, and dependence embodied in a body of \mathcal{L}_{cp} sentences as a system of

random variables. The main use of the time net is for answering queries about probabilities of facts and events over time. When a time net is created, only the marginal densities of root nodes are already known, perhaps along with some evidence about the value of some other random variables. Marginal densities for all other nodes must be computed on the basis of conditional densities and known marginal densities.

The algorithm that we show is a *sampling* algorithm. Sampling is a widely used method for estimating probability distributions, including its use for estimating probabilities in Bayesian networks (see, e.g., [Pearl, 1988]). Sampling is the best available general method for estimating densities for continuous variables, and it has been employed for continuous-time probabilistic reasoning [Berzuni *et al.*, 1989]. The algorithm given here is simpler than those previously proposed. It is a *forward* sampling method, suitable principally for predictive inference. Although more sophisticated algorithms are available, the algorithm is conceptually simple, and illustrates the essentials. The algorithm is easily extended to handle explanatory inference, for instance, by likelihood weighting [Shachter and Peot, 1991].

The central idea in sampling algorithms is to estimate variables, in our case densities, by repeated trials. In each trial, each random variable represented in a time net is *sampled* from its underlying distribution in a fixed order. A sample is a simulated value for the random variable. Each sample is *scored* according to some criteria. During repeated trials, a random variable takes different values with varying frequency, and the different values accumulate different scores. At the end, the scores are used to estimate the distribution for each random variable. Under certain conditions, the estimates are shown to converge to the true distribution (see, for example, [Geweke, to appear]).

For a discrete random variable, each sample is a particular discrete state out of a finite set. Thus, scoring the samples is easy. For a continuous random variable, we *discretize* the range of the continuous variable into a finite number of subranges, or *buckets*. A bucket is scored each time a sample falls within its subrange. The discretization defines the resolution to which we can distinguish the occurrence of events. In other words, it fixes the ϵ over which we considered an event to occur. We must bound the range in addition to discretizing it, to limit the number of buckets. At the same time, it is sometimes convenient to create distinguished uppermost and bottommost buckets containing $+\infty$ and $-\infty$, respectively. For a continuous range variable, we store the upper and lower bound of the sampled range and the score of each trial.

In our algorithm, we first find an order for sampling nodes. We may sample a node if it has no parents, or the values of all of its parents are known, either as evidence, or by sampling. It is easy to show that there is always such an order starting from any root

node, provided that the time net is a proper Bayesian network, *i.e.*, it is acyclic.

We sample each node from its distribution according to the order we find. In our basic algorithm, we score each sample equally by 1. At the end, we approximate the distribution of date nodes by examining the score in each bucket. Dividing by the number of trials gives the probability for that bucket.¹ This can be used to estimate a distribution such as $P(\text{beg}(\pi) \in [t, t+\epsilon])$. Because of our discretization, our estimate corresponds to the probability that $\text{beg}(\pi)$ occurred in the interval constituting the bucket that contains the time point t , not necessarily the probability that it occurred in the interval $[t, t+\epsilon]$. It is also possible to compute cumulative densities simply by summing; thus it is possible to answer queries about the probability that an event takes place over a longer interval, e.g., $P(\text{beg}(\pi) \in [u, v])?$, or even if it ever occurs at all, e.g., $P(\text{beg}(\pi) < +\infty)?$ (Note that we have loosely used an event for its date in these example queries).

In general, we do not compute the complete probability distribution for a range node. Instead, the probability of a particular range is computed only on demand. To do this, we count the number of trials whose sample is the queried range. For instance, in response to a query $P([8\text{am}, 10\text{am}] = \pi)?$, we count each trial where the sample for π was $[8\text{am}, 10\text{am}]$. The count is divided by the number of trials to estimate the probability. The query $P(\text{holds}(8\text{am}, 10\text{am}, \pi))?$ is different from the above, because $\text{holds}(8\text{am}, 10\text{am}, \pi)$ is true if π begins before 8 a.m. and ends after 10 a.m.. Thus, we would count each trial for which the sample for π contained the interval $[8\text{am}, 10\text{am}]$.

There is ample scope for optimizations and improvements to this scheme. We may apply variance reduction techniques such as importance sampling for faster convergence. In many cases, optimization depends heavily on knowing what type of queries are expected. If we are only interested in the marginal densities of a select few nodes, then we can considerably improve performance by reducing the network through node elimination [Kanazawa and Dean, 1989], and by focusing only on relevant portions of a network based on graph theoretic criteria [Shachter and Peot, 1991]. Although we have only considered sampling algorithms here, it may also be possible to further improve performance by combining with other types of algorithms.

Summary

We have described an approach of reasoning about time and probability using logic and time nets. The ideas and algorithms in this paper have been implemented in a program called *goo* (for *grandchild*

¹If we know the number of trials n beforehand, then we can score each sample by $1/n$ instead of 1. Alternatively, we can forget division altogether; the scores are proportional to the true distribution.

of ODDS; ODDS is an earlier probabilistic temporal database and decision theoretic planning system [Dean and Kanazawa, 1989b]). We are currently reimplementing goo as a probabilistic temporal database.

The probabilistic temporal database is a system for maintaining and applying knowledge and information about time and probability. It allows specification of general knowledge about facts, events, time, and probability in terms of \mathcal{L}_{cp} sentences. On the basis of this knowledge, the database constructs and incrementally augments or clips time nets in response to information about true events and relevant scenarios. It is thus similar to other systems that construct and maintain Bayesian networks on the fly (e.g., [Breese, 1987]).

As far as extensions to this work, promising extensions to the logic and to time nets involve continuous quantities, especially in the context of a time-branching logic of plans. Such an extension is useful for decision theoretic planning and for control applications [Haddawy, 1990]. It may also be applied fruitfully in control of inference [Dean and Boddy, 1988]. Other, relatively trivial, but useful, extensions include a life operator denoting the duration of fact tokens, and sentence formers for higher moments such as the mean.

Important outstanding research issues involve the construction of time nets from \mathcal{L}_{cp} sentences, including the automatic unfolding of circularities, and practical performance characteristics in large scale applications.

References

- Fahiem Bacchus. *Representating and Reasoning with Probabilistic Knowledge*. PhD thesis, University of Alberta, 1988. Also issued as Waterloo University Technical Report CS-88-31.
- Carlo Berzuni, Riccardo Ballazzi, and Silvana Quaglini. Temporal reasoning with probabilities. In *Proceedings of the Fifth Workshop on Uncertainty in Artificial Intelligence*, pages 14–21, Detroit, Michigan, August 1989.
- Carlo Berzuni. A probabilistic framework for temporal reasoning. *Artificial Intelligence*, to appear.
- John S. Breese. Knowledge representation and inference in intelligent decision systems. Technical Report 2, Rockwell International Science Center, 1987.
- D. R. Cox and D. Oakes. *Analysis of Survival Data*. Wiley, 1984.
- Thomas Dean and Mark Boddy. An analysis of time dependent planning. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 49–54, Minneapolis, Minnesota, 1988. AAAI.
- Thomas Dean and Keiji Kanazawa. Probabilistic temporal reasoning. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 524–528, Minneapolis, Minnesota, 1988. AAAI.
- Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, August 1989.
- Thomas Dean and Keiji Kanazawa. Persistence and probabilistic projection. *IEEE Transactions on Systems, Man and Cybernetics*, 19(3):574–585, May/June 1989.
- Luc Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, New York, 1986.
- J. Geweke. Bayesian inference in econometric models using monte carlo integration. *Econometrica*, to appear.
- Peter Haddawy. Time, chance, and action. In *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, pages 147–154, Cambridge, Massachusetts, 1990.
- Joseph Y. Halpern. An analysis of first-order logics of probability. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1375–1381, Detroit, Michigan, 1989. IJCAI.
- Steven John Hanks. *Projecting Plans for Uncertain Worlds*. PhD thesis, Yale University Department of Computer Science, January 1990.
- Keiji Kanazawa and Thomas Dean. A model for projection and action. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, Michigan, 1989. IJCAI.
- Keiji Kanazawa. *Probability, Time, and Action*. PhD thesis, Brown University, Providence, RI, Forthcoming.
- John McCarthy and Patrick J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 4, 1969.
- Drew V. McDermott. A temporal logic for reasoning about processes and plans. *Cognitive Science*, 6:101–155, 1982.
- Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
- Ross D. Shachter and Mark A. Peot. Simulation approaches to general probabilistic inference on belief networks. In John F. Lemmer and Laveen F. Kanal, editors, *Uncertainty in Artificial Intelligence 5*. North-Holland, 1991.
- Glenn Shafer and Judea Pearl, editors. *Readings in Uncertain Reasoning*. Morgan Kaufmann, Los Altos, California, 1990.
- Yoav Shoham. *Reasoning About Change: Time and Causation from the Standpoint of Artificial Intelligence*. MIT Press, Cambridge, Massachusetts, 1988.
- Jay Weber. *Principles and Algorithms for Causal Reasoning with Uncertainty*. PhD thesis, University of Rochester Computer Science, May 1989. Technical Report 287.