Generalizing Continuous Time Bayesian Networks with Immediate Nodes

Luigi Portinale and **Daniele Codetta-Raiteri**
Department of Computer Science
University of Piemonte Orientale "A. Avogadro"
{portinal, raiteri}@di.unipmn.it

Abstract

An extension to Continuous Time Bayesian Networks (*CTBN*) called Generalized *CTBN* (*GCTBN*) is presented; the formalism allows one to model, in addition to continuous time delayed variables (with exponentially distributed transition rates), also non delayed or "immediate" variables, which act as standard chance nodes in a Bayesian Network. The usefulness of this kind of model is discussed through an example concerning the reliability of a simple component-based system. A semantic model of *GCTBN*s, based on the formalism of Generalized Stochastic Petri Nets (*GSPN*) is outlined, whose purpose is twofold: to provide a well-defined semantics for *GCTBN*s in terms of the underlying stochastic process, and to provide an actual mean to perform inference (both prediction and smoothing) on *GCTBN*s. The example case study is then used, in order to highlight the exploitation of *GSPN* analysis for posterior probability computation on the *GCTBN* model.

1 Introduction

Temporal probabilistic graphical models allow for a factorization of the state space of a process, resulting in better modeling and inference features. Such models are usually based on graph structures, grounded on the theory of Bayesian Networks (*BN*). When time is assumed to be discrete, Dynamic Bayesian Networks (*DBN*) [7; 10] can be adopted. However, a discrete time assumption is not always adequate; for these reasons, Continuous Time Bayesian Networks (*CTBN*) have been proposed in [11; 12] and refined in [14]. Extensions to the basic model have also been proposed both regarding the use of indirect graph models [4] and the use of Erlang-Coxian distributions on the transition time [6].

In this paper, we propose another kind of extension and, in particular, a generalization of the standard *CTBN* framework, by allowing the presence of nodes which have no explicit temporal evolution; the values of such nodes are, in fact, "immediately" determined, depending on the values of other nodes in the network. The resulting framework is called Generalized *CTBN* (*GCTBN*) and is formally presented in Sec. 2. *GCTBN*s allow the modeling of processes having both a continuous-time temporal component and an immediate component capturing the logical/probabilistic interactions among modeled variables. While these modeling features are actually possible in discrete time *DBN*s, our work is, at the best of our knowledge, the first attempt trying to mix in the same *BN*, continuous-time delayed nodes with standard chance nodes.

In case of continuous time, a model having similar features can be found in the framework of Petri Nets, namely Generalized Stochastic Petri Nets (*GSPN*) [1][1] . Briefly, *GSPN*s are stochastic Petri nets, with two different sets of transitions, namely *temporal* with an exponentially distributed delay, and *immediate* transitions (with no delay), having priority over temporal ones. We propose to express a *GCTBN* model in terms of a *GSPN*, by means of a set of translation rules (see [13] for details). This translation is twofold: (1) it provides a well-defined semantics for a *GCTBN* model, in terms of the underlying stochastic process it represents (this is discussed in Sec. 4); (2) it provides an actual mean to perform inference on the *GCTBN* model, by exploiting well-studied analysis techniques for *GSPN*s, as described in Sec. 5.

Actually, in case of a *CTBN* exact inference may often be impractical, so approximations through message-passing algorithms on cluster graphs [12; 14], or through sampling [4; 5] have been proposed. In the present work, we take advantage of the correspondence between *GCTBN* and *GSPN*, in order to propose inference algorithms for *GCTBN* models (both for prediction and smoothing), based on *GSPN* solution algorithms and providing the exact solution of the model.

The possibilities offered by *GCTBN*s, can be exploited in several applications. For example, in system reliability analysis, it is very practical to distinguish between system components (having a temporal evolution) and specific modules or subsystems, whose behavior has to be modeled for the analysis. For instance, in Fault Tree Analysis (*FTA*), basic events represents the system components with their failure rates, while non-basic events are logical gates identifying modules of the system under examination [15]. In Dynamic Fault Trees [3], logical gates identifying sub-modules, can be combined with dynamic gates, modeling time-dependent dependencies (usually assuming continuous time) among com-

[1] Because of space restrictions, we refer the interested reader to [1; 13] for details and formal definitions.

ponents or sub-modules. Also in this case, it is very important to distinguish, at the modeling level, between delayed and immediate entities. Of course, similar considerations apply in other tasks as well, as in medical diagnosis, financial forecasting, biological process modeling, etc. Sec. 3 provides a simple case study in the reliability field, supporting the presentation of the concepts in the following sections.

2 The generalized *CTBN* model

Following the original paper in [11], a *CTBN* is defined as follows:

Definition 2.1 *Let $X = X_1, \ldots, X_n$ be a set of discrete variables, a* CTBN *over X consists of two components. The first one is an initial distribution P_X^0 over X (possibly specified as a standard* BN *over X). The second component is a continuous-time transition model specified as (1) a directed graph G whose nodes are $X_1, \ldots, X_n$ (and with $Pa(X_i)$ denoting the parents of X_i in G); (2) a conditional intensity matrix $Q_{X_i|Pa(X_i)}$ for every $X_i \in X$.*

We can now introduce the notion of a Generalized *CTBN* (*GCTBN*).

Definition 2.2 *Given a set of discrete variables $X = \{X_1, \ldots, X_n\}$ partitioned into the sets D (delayed variables) and I (immediate variables) (i.e. $X = D \cup I$ and $D \cap I = \emptyset$), a Generalized Continuous Time Bayesian Network (*GCTBN*) is a pair $N = \langle P_X^0, G\rangle$ where*

- *P_X^0 is an initial probability distribution over X;*
- *G is a directed graph whose nodes are $X_1, \ldots, X_n$ (and with $Pa(X_i)$ denoting the parents of X_i in G) such that*
  1. *there is no directed cycle in G composed only by nodes in the set I;*
  2. *for each node $X \in I$ a conditional probability table $P[X|Pa(X)]$ is defined (as in standard* BN*);*
  3. *for each node $Y \in D$ a conditional intensity matrix $Q_{Y|Pa(Y)}$ is defined (as in standard* CTBN*).*

Delayed (or temporal) nodes are, as in case of a *CTBN*, nodes representing variables with a continuous time evolution ruled by exponential transition rates, and conditioned by the values of parent variables (that may be either delayed or immediate). Immediate nodes are introduced, in order to capture variables whose evolution is not ruled by transition rates associated with their values, but is conditionally determined at a given time point, by other variables in the model. Such variables are then treated as usual chance nodes in a *BN* and have a standard Conditional Probability Table (*CPT*) associated with them.

A few words are worth to be spent for the structure of the graph modeling the *GCTBN*. While it is in general possible to have cycles in the graph (as in *CTBN*) due to the temporal nature of some nodes, such cycles cannot be composed only by immediate nodes. Indeed, if this was the case, we would introduce static circular dependencies among model variables.

Finally, it is worth noting that the initial distribution P_X^0 can in general be specified only on a subset of X. In particular, let $R \subset I$ be the set of root nodes (i.e. node with no parent in G) which are immediate, then the initial distribution can be computed as $P_X^0 = P_{R\cup D}^0 \prod_{Y_j \in (I-R)} P[Y_j|Pa(Y_j)]$. In fact, while it is necessary to specify an initial distribution over delayed variables, the distribution on the immediate variables can be determined depending on the values of their parents; of course if an immediate variable is modeled as a root node, an initial prior probability is needed[2].

3 An illustrative example

We now consider a case study which can be easily modeled in form of *GCTBN*. This is a typical case in the field of reliability analysis, and consists of a small system composed by the main component A and its "warm" spare component B. This means that initially both components are working, but A is active while B is dormant; in case of failure of A, B is activated in order to replace A in its function. We assume that the activation of B occurs with a 0.99 probability. If B fails before A, B can not replace A.

The system is considered as failed if A is failed and B is dormant or failed. We suppose that only while the system is failed, the components A and B undergo repair. As soon as the repair of one of the components is completed, the component re-starts in working state: if A is repaired the system becomes operative again and the repair of B is suspended; if instead B is repaired, this may determine one of these two situations: 1) B may become active with probability $p = 0.99$ and consequently the system becomes operative again and the repair of A is suspended. 2) B may become dormant with probability $1 - p$, so the system is still failed and the repair of B goes on.

The component time to failure or repair is a random variable ruled by the negative exponential distribution according to the component failure or repair rate respectively. In the case of the main component A, the failure rate is λ_A =1.0E−06 h^{-1}. The failure rate of B, λ_B, changes according to its current state: if B is dormant, λ_B is equal to 5.0E−07 h^{-1}; if instead B is active, λ_B is equal to 1.0E−06 h^{-1}. Because of this, the spare is defined as "warm" [3]. A and B have the same repair rate: μ_A =μ_B =0.01 h^{-1}.

3.1 The *GCTBN* model

The case study described above is represented by the *GCTBN* model in Fig. 1 where the variables A, B, SYS represent the state of the component A, the component B and the whole system respectively. All the variables are binary because each entity can be in the working or in the failed state (for the component B, the working state comprises both the dormancy and the activation). In particular, we represent the working state with the value 1, and the failed state with the value 2.

The variable A influences the variable B because the failure rate of the component B depends on the state of A. Both the variables A and B influence the variable SYS because the

[2]Actually, since prior probabilities on immediate root nodes are a special case of CPT, we could also simply write $P_X^0 = P_D^0 \prod_{Y_j \in I} P[Y_j|Pa(Y_j)]$, to emphasize the fact that, for the specification of the temporal evolution of the model, the only initial distribution is on delayed nodes (the other parameters are actually a fixed specification on the network).

state of the whole system depends on the state of the components A and B. The arcs connecting the variable SYS to A and B respectively, concern the repair of the components A and B: only while the system is failed, they can be repaired.

The variables A and B in the *GCTBN* model in Fig. 1 are delayed variables (Sec. 2) and are drawn as double-circled nodes: both variables implicitly incorporate a Continuous Time Markov Chain (*CTMC*) composed by two states: 1 (working) and 2 (failed). Due to the assumption that both components are initially supposed to work, the initial probability distribution is set equal to 1 for states $A = 1$ and $B = 1$. In the case of A, the current value of the rates λ_A and μ_A depends on the current value of the variable SYS, the only one influencing A. This is shown by the Conditional Intensity Matrix (*CIM*) reported in Tab. 1.a, where we can notice that the rate μ_A is not null only if the value of SYS is 2. The rate λ_A instead, is constant.

In the case of the variable B, the current value of the rates λ_B and μ_B depends on the current value of the variables A and SYS, as shown by the *CIM* appearing in Tab. 1.b, where λ_B is increased only when A is equal to 2 and SYS is equal to 1 (this implies that B is active). As in the case of the variable A, the rate μ_B is not null only if the value of SYS is 2. Notice that the combination $A = 1, SYS = 2$ is impossible, so the corresponding entries are not significant.

The variable SYS is immediate (Sec. 2) and is shown as a circle node in Fig. 1. It is characterized by the *CPT* appearing in Tab. 1.c. In particular, SYS is surely equal to 1 if A is equal to 1, and surely equal to 2 if both A and B are equal to 2. In the case of A equal to 2 and B equal to 1, SYS assumes the value 1 with probability 0.99 (this implies the activation of the spare component B), or the value 2 with probability 0.01 (this implies that B is still dormant).

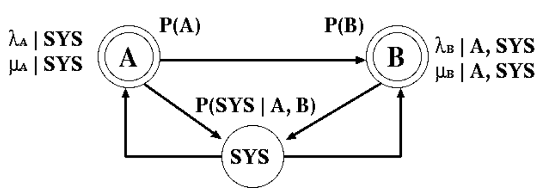

Figure 1: *GCTBN* model of the case study.

a)

	1		2	
1			SYS	λ_A
			1	1.0E-06 h^{-1}
			2	1.0E-06 h^{-1}
2	SYS	μ_A		
	1	0 h^{-1}		
	2	0.01 h^{-1}		

b)

	1			2		
1				A	SYS	λ_B
				1	1	5.0E-07 h^{-1}
				1	2	–
				2	1	1.0E-06 h^{-1}
				2	2	5.0E-07 h^{-1}
2	A	SYS	μ_B			
	1	1	0 h^{-1}			
	1	2	–			
	2	1	0 h^{-1}			
	2	2	0.01 h^{-1}			

c)

A	B	SYS	Prob.
1	1	1	1
1	1	2	0
1	2	1	1
1	2	2	0
2	1	1	0.99
2	1	2	0.01
2	2	1	0
2	2	2	1

Table 1: a) *CIM* for the variable A. b) *CIM* for the variable B. c) *CPT* for the variable SYS in the *GCTBN* model in Fig. 1.

4 A Petri Net semantics for *GCTBN*

The combination in a single model of entities explicitly evolving over time with entities whose determination is "immediate", has been already proposed in frameworks other than *CTBN*; as we have already noticed in Sec. 1, *DBN*s provide an example, in case of discrete time. In case of continuous time, *GSPN*s allow to model both kinds of entities by means of temporal and immediate transitions respectively. This means that, in case both an immediate and a temporal transition are enabled, the firing of the former takes precedence over the firing of the latter. Immediate transitions may also have different priority levels among them.

The stochastic process associated with a *GSPN* is a homogeneous continuous time semi-Markov process that can be analyzed either by solving the so called *Embedded Markov Chain* or by removing from the set of possible states, the so-called *vanishing states* or *markings* and by analyzing the resulting *CTMC* [1]. Vanishing states are the state (or markings) resulting from the firing of immediate transitions; they can be removed, since the system does not spend time in such states. This removal operation has also the advantage of reducing (often in a significant way) the set of possible states to be analyzed.

Solution techniques for *GSPN*s have received a lot of attention, especially with respect to the possibility of representing in a compact way the underlying *CTMC* and in solving it efficiently [8; 9]. Once a *GCTBN* has been compiled into a *GSPN* [13], such techniques can be employed to compute inference measures on the original *GCTBN* model (see Sec. 5).

There are two main analyses that can be performed with a *GSPN*: *steady state* and *transient analysis*. In the first case, the equilibrium distribution of the states is computed, while in the latter, such a distribution is computed at a given time point. In particular, solving a *GSPN* (for either steady state or transient analysis) can provide the probability distribution of the number of tokens in each place. This information can then be exploited, in order to perform inference on the original *GCTBN* model as it will be shown in Sec. 5.

4.1 The *GSPN* model for the case study

According to the conversion rules described in [13], the *GCTBN* of the case study in Fig. 1 can be converted into the *GSPN* model shown in Fig. 2 where the places A, B and SYS correspond to the variables in the *GCTBN* model. The value of a *GCTBN* variable is mapped into the marking (number of tokens) of the corresponding place in the *GSPN*. Let us consider the place B in the *GSPN*: the marking of the place B can be equal to 1 or 2, the same values that the variable B in the *GCTBN* can assume. B is a delayed variable and its initialization is modeled in the *GSPN* by the immediate transitions B_init_1 and B_init_2 called *"init" transition*. Such transitions are both initially enabled to fire with the effect of setting the initial marking of the place B to 1 or 2 respectively. The probability of these transitions to fire corresponds to the initial probability distribution of the variable B.

The variation of the marking of the place B is determined by the timed transitions B_1_2 and B_2_1. The transition B_1_2 is enabled to fire when the place B contains one token; the effect of its firing is setting the marking of B to 2. The transition B_2_1 instead, can fire when the marking of the place B is equal to 2, and turns it to 1. The dependency of the transition rate of a variable on the values of the other variables in the *GCTBN* model, becomes in the *GSPN* model, the dependency of the firing rate of a timed transition on the markings of the other places. For instance, in the *GCTBN* model the variable B depends on A and SYS; let us consider λ_B, the transition rate of B from 1 to 2 depending on the values of the variables A and SYS (Tab. 1.b). In the *GSPN* model, λ_B becomes the firing rate of the timed transition B_1_2 whose value depends on the marking of the places A and SYS, and assumes the same values reported in Tab. 1.b. The firing rate of the timed transition B_2_1 instead, will correspond to the rate μ_B reported in Tab. 1.b, still depending on the marking of the places A and SYS.

The initialization of the marking of the place A is modeled by the immediate init transitions A_init_1 and A_init_2, while the variation of its marking is modeled by the timed transitions A_1_2 and A_2_1, but in this case their firing rate will depend only on the marking of the place SYS, because in the *GCTBN* model the variable A depends only on the variable SYS. Such variable is immediate in the *GCTBN* and depends on A and B. Therefore in the *GSPN* each time the marking of the place A or of the place B is modified, the marking of SYS has to be immediately updated: each time the transition A_1_2, A_2_1, B_1_2 or B_2_1 fires, one token appears in the place $emptySYS$; this determines the firing of the immediate transition $reset\_SYS\_1$ or $reset\_SYS\_2$ having priority over the other immediate transitions (priority level $\pi = 2$ in Fig. 2), with the effect of removing any token inside the place SYS. At this point, the marking of such place has to be set according to the current marking of the places A and B. This is done by one of the immediate transitions set_SYS_1, set_SYS_2, set_SYS_3, set_SYS_4, set_SYS_5. Each of them corresponds to one entry having not null probability in the *CPT* of the variable SYS in the *GCTBN* model (Tab. 1.c). Each of such transitions has the same probability and the same effect on the marking of the place SYS, as the corresponding entry in the *CPT*.

5 Inference

Standard inference tasks in temporal probabilistic models are *prediction* and *smoothing* [10]. *Prediction* is the task of computing the probability of a set of queried variables, given past evidence, i.e. predicting a future state taking into consideration the observations up to now (a special case occurs when the last evidence time point and the query time are the same and is called *Filtering* or *Monitoring*). *Smoothing* is the task

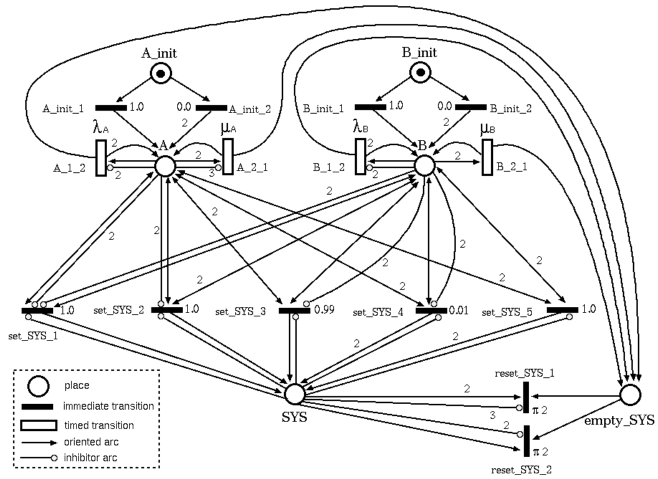

Figure 2: *GSPN* model obtained from the *GCTBN* in Fig. 1.

of estimating what happened $k > 0$ time points in the past, given all the evidence (observations) up to now. Such tasks can be accomplished, depending on the model adopted, by inference procedures usually based on specific adaptation of standard algorithms for Bayesian Networks. For instance, in *DBN* models, both exact algorithms based on junction tree [10] as well as approximate algorithms exploiting the net structure [2] or based on stochastic simulation can be employed. In this paper, we propose the conversion into *GSPN*, and the *GSPN* analysis methods, as means to compute exact inference on the *GCTBN* model, for both prediction and smoothing tasks.

Computing the probability of a given variable assignment $X = x_i$ at time t, will correspond to compute the probability of having i tokens in the place modeling X at time t. In particular, if $P()$ is the probability function associated with the *GCTBN* model and $Pr\{\}$ is the probability function associated with the *GSPN* model, then $P(X_t = x_i) = Pr\{\#X_t = i\}$, where X_t is the value of X at time t and $\#X_t$ is the number of tokens in the place corresponding to X at time t.

5.1 Prediction Inference

The task of prediction consists in computing the posterior probability at time t of a set of queried variables $Q \subseteq D \cup I$, given a stream of observations (evidence) $e_{t_1}, \ldots, e_{t_k}$ from time t_1 to time t_k with $t_1 < \ldots < t_k < t$. Every evidence e_{t_j} consists of a (possibly different) set of instantiated variables.

Prediction can then be implemented by repeatedly solving the transient of the corresponding *GSPN* at the observation and query times. Of course, any observation will condition the evolution of the model, so the suitable conditioning operations must be performed before a new *GSPN* resolution. The pseudo-code for the prediction procedure is shown in Fig. 3. Notice that, in the special case of *filtering*, the last evidence would be available at the query time (i.e. $t = t_k$ in Fig. 3); in such a case, the update of the transition weights (last statement in the `for` cycle) is not necessary, as well as the final transient solution. The procedure would then simply output

```
Procedure PREDICTION
INPUT: a set of queried variables Q, a query time t, a set of
temporally labeled evidences e_{t_1},...,e_{t_k}
with t_1 < ... < t_k < t
OUTPUT: P(Q_t|e_{t_1},...,e_{t_k})

let t_0 = 0;

for i = 1 to k {
   solve the GSPN transient at time (t_i - t_{i-1});
   compute from transient, p_i(j) = Pr{X_j|e_{t_i}} for X_j ∈ D ∪ R;
   update the weights of the immediate init transitions of X_j
     according to p_i(j); }
solve the GSPN transient at time (t - t_k);
compute from transient, r = Pr{Q};

output r;
```

Figure 3: The prediction inference procedure.

$Pr\{Q|e_t\}$ computed from the last transient analysis.

In case there is evidence available at time $t_0 = 0$, if the evidence is on variables $X \in D \cup R$, then it is incorporated into their "init" distribution; if the evidence is on variables $X \in I - R$, then the "init" of the other variables are updated by solving the transient at time $t_0 = 0$.

5.2 Smoothing Inference

The smoothing task consists in computing the probability at time t of a set of queried variables $Q \subseteq D \cup I$, given a stream of observations (evidence) $e_{t_1}, \ldots, e_{t_k}$ from time t_1 to time t_k with $t < t_1 < \ldots < t_k$. The issue is how to condition on variables observed at a time instant that follows the current one. The idea is then to try to reformulate the problem in such a way that it can be reduced to a prediction-like task. The approach is then based on the application of the Bayes rule as follows:

$$P(Q_t|e_{t_1}, \ldots, e_{t_k}) = \alpha P(Q_t) P(e_{t_1}, \ldots, e_{t_k}|Q_t)$$
$$= \alpha P(Q_t) P(e_{t_1}|Q_t) \ldots P(e_{t_k}|e_{t_1}, \ldots, e_{t_{k-1}}, Q_t)$$

In this way, every factor in the above formula is conditioned on the past and can be implemented as in prediction. However, the computation of the normalization factor α, requires that a separate computation must be performed for every possible assignment of the query Q. The interesting point is that such computations are independent, so they can be possibly performed in parallel[3]. Once the computation has been performed for every query assignment, then results can be normalized to get the actual required probability values.

The pseudo-code for the smoothing procedure is shown in Fig. 4. The `normalize` operator, just divide any entry of the vector A by the sum of all the entries, in order to provide the final probability vector of the query.

5.3 Example of inference in the case study

Consider again the case study of Fig. 1. Concerning prediction, let us consider to observe the system working ($SYS = 1$) at time $t = 10^5 h$ and the system failed ($SYS = 2$) at time $t = 2 \cdot 10^5 h$. By considering the procedure outlined in Fig. 3 we can compute the probability of component A being working at time $t = 5 \cdot 10^5 h$, conditioned by the observation stream, as follows: (1) we solve the transient at $t = 10^5 h$ and we compute the probabilities of A and B, conditioned by the observation $SYS = 1$; (2) we use the above computed probabilities as the new init probabilities for the places A and B of the *GSPN*; (3) we solve the transient for another time interval $t = 10^5 h$ and we compute the probabilities of A and B, conditioned by the observation $SYS = 2$; (4) we use the above computed probabilities as the new init probabilities for the places A and B of the *GSPN*; (5) we solve the transient for a time interval $t = 3 \cdot 10^5 h$ and we finally compute the probability of the query A. Tab. 2 shows the values computed during the above process. The last row shows the required results.

[3] An alternative can be to directly compute the denominator of the Bayes formula (i.e. the probability of the evidence stream); however, this requires a larger number of transient solutions if the length of the observation stream is greater than the the number N of assignments of Q (i.e.if $k > N$), as is usually the case.

```
Procedure SMOOTHING
INPUT: a set of queried variables Q, a query time t, a set of
temporally labeled evidences e_{t_1},...,e_{t_k}
with t < t_1 < ... < t_k
OUTPUT: P(Q_t|e_{t_1},...,e_{t_k});

let N be the cardinality of possible assignments q_i (1≤i≤N) of Q;
A: array[N];
for i = 1 to N {
  //possibly in parallel
  A[i]=SMOOTH(q_i); }
output normalize(A);

Procedure SMOOTH(q) {
   t_0 = t;
   solve the GSPN transient at time t;
   compute from transient, r = Pr{Q = q};
   ev = q;
   for i = 1 to k {
     compute from transient, p_{i-1}(j) = Pr{X_j|ev} for X_j ∈ D ∪ R;
     update the weights of the immediate init transitions of X_j
       according to p_{i-1}(j);
     solve the GSPN transient at time (t_i - t_{i-1});
     compute from transient, p_i(e) = Pr{e_{t_i}};
     r = r p_i(e);
     ev = e_{t_i}; }
   output r; }
```

Figure 4: The smoothing inference procedure.

Time (h)	$P(A=1\|e)$	$P(A=2\|e)$	$P(B=1\|e)$	$P(B=2\|e)$
100000	0.909228	0.090772	0.952445	0.047555
200000	0	1	0.071429	0.928571
500000	**0.521855**	**0.478145**	-	-

Table 2: Probabilities for prediction inference in the case study (e is the current accumulated evidence).

Concerning smoothing inference, let us suppose to have observed the system working at time $t = 3 \cdot 10^5 h$ and failed at time $t = 5 \cdot 10^5 h$. We ask for the probability of component A at time $t = 2 \cdot 10^5 h$, conditioned by the above evidence. By considering the procedure outlined in Fig. 4 we can compute the required probabilities as follows: (1) we first consider the case $A = 1$; (2) we solve the transient at $t = 2 \cdot 10^5 h$ and we compute $r1 = P(A = 1)$; (3) we condition A and B on $A = 1$ and we determine the new init probabilities for A and B; (4) we solve the transient for $t = 10^5 h$ (to reach time $3 \cdot 10^5 h$) and we compute $r2 = P(SYS = 1)$; we also condition A and B on $SYS = 1$ and we use such values as new init probabilities for places A and B; (5) we solve the transient for $t = 2 \cdot 10^5 h$ (to reach time $5 \cdot 10^5 h$) and we compute $r3 = P(SYS = 2)$; (6) we compute the un-normalized probability of $A = 1$ as $p1 = r1 \cdot r2 \cdot r3$; By performing the above steps also for the case $A = 2$ we can similarly compute the un-normalized probability of $A = 2$, namely $p2$. A simple normalization over $p1$ and $p2$ will then produce the required results. Tab. 3 shows the values computed during the above process (partial results $r1, r2, r3$ are shown in bold).

$P(A = 1)$ at $t = 2 \cdot 10^5$ = **0.833086**

Time (h)	$P(A = 1\|e)$	$P(B = 1\|e)$	$P(SYS = 1\|e)$	$P(SYS = 2\|e)$
200000	1	0.891238	-	-
300000	0.913981	0.854028	**0.999988**	-
500000	-	-	-	**0.000022**

p1=0.0000183277

$P(A = 2)$ at $t = 2 \cdot 10^5$ = **0.166914**

Time (h)	$P(A = 1\|e)$	$P(B = 1\|e)$	$P(SYS = 1\|e)$	$P(SYS = 2\|e)$
200000	0	0.999922	-	-
300000	0.056648	0.952429	**0.999950**	-
500000	-	-	-	**0.000049**

p2=0.0000081784

$P(A = 1\|e)$	$\frac{p1}{p1+p2}$=**0.691452**
$P(A = 2\|e)$	$\frac{p2}{p1+p2}$=**0.308548**

Table 3: Probabilities for smoothing inference in the case study (e is the current accumulated evidence).

6 Conclusions and Future Works

In this paper we have presented a generalized *CTBN* formalism, allowing one to mix in the same model continuous time delayed variables with standard "immediate" chance variables. The usefulness of this kind of model has been discussed through an example concerning the reliability of a simple component-based system. The semantics of the proposed *GCTBN* formalism has been provided in terms of Generalized Stochastic Petri Nets (*GSPN*), a well-known formalism isomorph to semi-Markov processes, through which it is also possible to exploit well established analysis techniques, in order to perform standard prediction or smoothing inference. In particular, adopting *GSPN* solution algorithms as the basis for *GCTBN* inference, allows one to take advantage of specialized methodologies for solving the underlying stochastic process, that are currently able to deal with extremely large models; in particular, such techniques (based on data structures like matrices or decision diagrams) allow for one order of magnitude of increase in the size of the models to be solved exactly, with respect to standard methods, meaning that models with an order of 10^{10} tangible states can actually be solved [8; 9].

However analyzing a *GCTBN* by means of the underlying *GSPN* is only one possibility that does not take explicit advantage of the structure of the graph as in *CTBN* algorithms [12; 14]. Our future works will try to investigate the possibility of adopting cluster-based or stochastic simulation approximations, even on *GCTBN* models, and in comparing their performance and quality with respect to *GSPN*-based solution techniques. In particular, since Petri nets are a natural framework for event-based simulation, it would be interesting to investigate how simulation-based approximations can be actually guided by the underlying *GSPN* model. Finally, since symbolic representations (based on matrices or decision diagrams) have been proved very useful for the analysis of *GSPN* models, it would also be of significant interest to study the relationships between such representations and the inference procedures on probabilistic graphical models in general, since this could in principle open the possibility of new classes of algorithms for *BN*-based formalisms.

References

[1] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets*. J. Wiley, 1995.

[2] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *Proceedings UAI 1988*, pages 33–42, 1998.

[3] J. Bechta Dugan, S.J. Bavuso, and M.A. Boyd. Dynamic fault-tree models for fault-tolerant computer systems. *IEEE Trans. on Reliability*, 41:363–377, 1992.

[4] T. El-Hay, N. Friedman, and R. Kupferman. Gibbs sampling in factorized continuous time Markov processes. In *Proc. 24rd UAI'08*, 2008.

[5] Y. Fan and C. Shelton. Sampling for approximate inference in continuous time Bayesian networks. In *Proc. 10th Int. Symp. on AI and Mathematics*, 2008.

[6] K. Gopalratnam, H. Kautz, and D.S. Weld. Extending continuous time Bayesian networks. In *Proc. AAAI'05*, pages 981–986, Pittsburgh, PA, 2005.

[7] U. Kjaerulff. dHugin: a computational system for dynamic time-sliced Bayesian networks. *International Journal of Forecasting*, 11:89–101, 1995.

[8] A.S. Miner. Decision diagrams for the exact solution of Markov models. *Proceedings in Applied Mathematics and Mechanics (PAMM)*, 7(1), 2007.

[9] A.S. Miner and D. Parker. Symbolic representation and analysis of large probabilistic systems. In *Validation of Stochastic Systems, LNCS 2925*, pages 296–338. Springer, 2004.

[10] K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD Thesis, UC Berkley, 2002. http://www.cs.ubc.ca/ murphyk/Thesis/thesis.html.

[11] U. Nodelman, C.R. Shelton, and D. Koller. Continuous Time Bayesian Networks. In *Proc. 18th UAI'02*, pages 378–387, 2002.

[12] U. Nodelman, C.R. Shelton, and D. Koller. Expectation propagation for continuous time Bayesian networks. In *Proc. 21st UAI'05*, pages 431–440, 2005.

[13] L. Portinale and D. Codetta-Raiteri. A GSPN semantics for continuous time Bayesian networks with immediate nodes. Technical Report TR-INF-2009-03-03-UNIPMN, Computer Science Dept., UPO, 2009. http://www.di.unipmn.it/TechnicalReports/TR-INF-2009-03-03-UNIPMN.pdf.

[14] S. Saria, U. Nodelman, and D. Koller. Reasoning at the right time granularity. In *Proc. 23rd UAI'07*, pages 421–430, 2007.

[15] W. G. Schneeweiss. *The Fault Tree Method*. LiLoLe Verlag, 1999.