From Dynamic Influence Nets to Dynamic Bayesian Networks: A Transformation Algorithm

Sajjad Haider[1,2,*]
[1]Center for Computer Studies, Institute of Business Administration, Karachi 74400, Pakistan
[2]IBA City Campus, Garden Road, Karachi 74400, Pakistan

This paper presents an algorithm to transform a dynamic influence net (DIN) into a dynamic Bayesian network (DBN). The transformation aims to bring the best of both probabilistic reasoning paradigms. The advantages of DINs lie in their ability to represent causal and time-varying information in a compact and easy-to-understand manner. They facilitate a system modeler in connecting a set of desired effects and a set of actionable events through a series of dynamically changing cause and effect relationships. The resultant probabilistic model is then used to analyze different courses of action in terms of their effectiveness to achieve the desired effect(s). The major drawback of DINs is their inability to incorporate evidence that arrive during the execution of a course of action (COA). Several belief-updating algorithms, on the other hand, have been developed for DBNs that enable a system modeler to insert evidence in dynamic probabilistic models. Dynamic Bayesian networks, however, suffer from the intractability of knowledge acquisition. The presented transformation algorithm combines the advantages of both DINs and DBNs. It enables a system analyst to capture a complex situation using a DIN and pick the best (or close-to-best) COA that maximizes the likelihood of achieving the desired effect. During the execution, if evidence becomes available, the DIN is converted into an equivalent DBN and beliefs of other nodes in the network are updated. If required, the selected COA can be revised on the basis of the recently received evidence. The presented methodology is applicable in domains requiring strategic level decision making in highly complex situations, such as war games, real-time strategy video games, and business simulation games.

1. INTRODUCTION

During the last two decades, Bayesian networks (BNs) have emerged as the most popular tool for modeling and reasoning in uncertain domains. They have been successfully applied in several domains including medical diagnosis, terrorism, forecasting, information fusion, system troubleshooting, etc. Mittal and Kassim[1]

*Author to whom all correspondence should be addressed: e-mail: sajjad.haider@khi.iba.edu.pk.

INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS, VOL. 24, 919–933 (2009)
 Published online in Wiley InterScience (www.interscience.wiley.com). • DOI 10.1002/int.20367

and Pourret et al.[2] provide an extensive list of areas in which BNs have been applied. A BN is a graphical representation of a joint probability distribution. It consists of two components. The first is a directed acyclic graph in which each node represents a random variable, whereas the set of arcs connecting pairs of nodes represents certain conditional independence properties. This component captures the *structure* of the probability distribution. The second component is a collection of *parameters* that describe the conditional probability of each variable, given its parent in the graph. Together, these two components represent a unique probability distribution.[3]

Bayesian networks were originally designed to capture *static* interdependencies among variables in an uncertain situation. The last few years have seen an emergence of techniques that attempt to integrate the notion of time and uncertainty. The most popular of them is called dynamic Bayesian network (DBN).[4] A DBN is created by discretizing time and creating instances of variables in a BN for each point in the time interval under consideration. Having its roots in the canonical BN, however, a DBN suffers from the same limitation as a BN, that is, (a) intractability of inference and (b) intractable elicitation of all the conditional probabilities. The first issue deals with computing the likelihood of variables of interest in reasonable amount of time, whereas the second issue deals with modeling complex situations using minimum amount of information. Several attempts have been made to address both issues. Approximate and simulation-based algorithms have been proposed that exploit certain conditions in a DBN to efficiently compute the likelihood of variables of interest.[5–9] Efforts have also been made to add different types of temporal constructs to the existing BN-based formalism to enhance its modeling capabilities.[10–16]

Influence nets (INs), a special instance of BNs, were developed with the aim of overcoming these two limitations of BNs, that is, intractability of knowledge elicitation and intractability of inference. They use the CAST logic,[17,18] a variant of the noisy-OR[19,20] approach, for knowledge elicitation, whereas the belief propagation is done with the independence of parents' assumption (also known as loopy-belief propagation[21]). The main advantage of INs lies in their compact and easy-to-understand representation, especially in domains in which it is hard to find empirical data. The CAST logic–based knowledge acquisition mechanism allows modeling of negative and positive influences. These influences are ultimately transformed into conditional probability tables (CPTs), but they immensely ease the knowledge acquisition process.

Enhancements were made by Wagenhals and Wentz.[22] that allow an IN to capture information and processing delays by associating time delays with arcs and nodes, respectively. The enhanced model also allows the provision to associate time-stamps with the actionable events. These time-stamps specify the time at which these actions are taken. Influence nets with these extensions are referred to as timed influence nets (TINs). Recently, Haider and Levis[23] have further extended the IN-based formalism and have added the concepts of time-varying (nonstationary) influences and self-loop. A self-loop models the dependency of the current state of an event on its previous state. An IN with all these temporal extensions is called a dynamic influence net (DIN).

These time-dependent INs allow a system analyst to observe the changes in the probability of a particular event over a period of time. They have been experimentally used in the area of effects-based operations for evaluating alternate courses of action and their effectiveness to a mission's objectives. Wentz and Wagenhals.[24] developed a TIN to model the political crisis that occurred in East Timor during the final years of the previous decade. The model was developed as a prototype for the Decision Support System for Coalition Operations developed by SPAWAR Systems Center, San Diego, to support the Operations Planning Team of the Commander in Chief, U.S. Pacific Command. Wagenhals et al.[25] developed a TIN to assess a nation's ability to wage war on the basis of damage effects to its civil infrastructure. Wentz and Wagenhals[24] developed a TIN to model broad-front, national-level actions needed to achieve an outcome that deterred a terrorist field cell from attacking. DeGregorio et al.[26] developed a TIN to model certain aspect of first Gulf war. Wagenhals and Wentz[22] developed a TIN to model chemical and biological threat from terrorists. Wagenhals and Levis[27] developed a TIN to evaluate a complex situation in which an adversary is embedded in a society from which it is receiving support.

All of these time-dependent INs were developed with the aim of identifying the best course of action (COA) under complex uncertain situations. This process of best COA identification using manual trial and error method is an extremely difficult task because it requires exploring millions (or even more) of possible options. Haider and Levis[28,29] have developed evolutionary algorithms and particle swarm optimization–based approaches to automate this extremely difficult process of effective COA identification. Their approach can also handle certain types of causal and temporal constraints that might be present among actionable events in a problem domain.

A DIN provides a compact and intuitive knowledge elicitation framework to model dynamic uncertain situations. In contrast to an exponential number of parameters required for the specification of a DBN (especially in the case of nonstationary conditional probabilities), a DIN requires only a linear number of parameters for model specification. The DIN framework, however, is mainly suitable for COA evaluation and is not designed to incorporate evidence that arrive during the execution of a COA. Several algorithms, on the other hand, have been developed for belief updating in a DBN. This paper synergizes the capabilities of both modeling paradigms and presents a transformation algorithm that converts a DIN into a DBN and uses both models during different stages of model building and decision making. The model building part and effective COA(s) selection is done with DIN. During the execution of the COA, if evidence becomes available, then the DIN is converted into a DBN and belief updating is performed with standard DBN belief updating algorithms. The work is an extension of an earlier work by Haider and Zaidi[30] that transforms a TIN into a time-sliced BN. The presented methodology is applicable in complex domains that require high-level strategic decision making and that rely heavily on subjective information. This includes war games, real-time strategy video games, business simulation games, etc. The rest of paper is organized as follows. Section 2 explains the CAST logic and DINs, whereas DBNs are explained in Section 3. The transformation of a DIN into a DBN is explained in Section 4. Finally, Section 5 concludes the paper and discusses the future research direction.

2. INFLUENCE NET–BASED FORMALISM

2.1. CAST Logic

A DIN is a graph-based probabilistic approach to capture uncertainty in dynamically changing environment. It has its origin in INs, which, as a special instance of BNs, were designed to capture static situations. Influence nets differ with BNs in the way knowledge is acquired from the experts. Instead of asking the complete CPTs, INs allow a subject matter expert to specify the influence of an event (parent node) on another connected event (child node) using the CAST logic parameters. The logic is an extension of the noisy-OR approach. The CAST logic requires two values for each link in an IN. The first represent the impact of parent node being true on the child node, whereas the second represents the impact of parent node being false on the child node. The parameters take values in the range of $(-1, 1)$. The negative values show a negative influence of an event on its child event, whereas the positive values show a positive influence of an event on its child event. The ability of INs to model both positive and negative influences in an intuitive manner has been found extremely useful in modeling domains relying heavily on subjective information. Once a user defines all the parameters, they are converted into CPTs and the resultant tables are used during the probability propagation phase. Thus, an exponential number of values are general from linear number of parameters. The four major steps of the logic are briefly explained in the following text with the help of Figure 1.[23] Readers interested in detailed working of the logic should refer to Rosen and Smith[18] and Chang et al.[17]

Figure 1 contains four nodes A, B, C, and X. On each arc, two causal strengths are specified. These numbers represent the probability that a specified state of a parent node will cause a certain state in the child node. Positive values on arcs are causal influences that cause a node to occur with some probability, whereas negative values are influences that cause the negation of a node to occur with some probability. For instance, the arc between B and X has values between -0.4 and 0.8. The first value, referred to as h, states that if B is true, then this will cause X to be false with probability 0.4, whereas the second value, referred to as g, states that if B is false, then this will cause X to be true with probability 0.8. Both h and g can take values in the interval $(-1, 1)$. All nonroot nodes are assigned a baseline probability, which is similar to the "leak" probability in the noisy-OR approach. This probability is the user-assigned assessment that the event would occur independently of the modeled

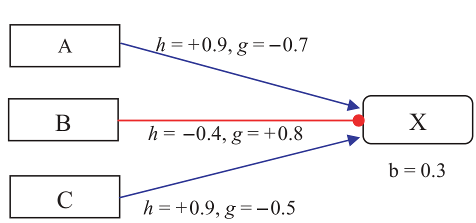

Figure 1. An influence network with CAST logic parameters.

influences in a net. There are four major steps in the CAST logic algorithm that converts the user-defined parameters into conditional probabilities:

(a) aggregate positive causal strengths,
(b) aggregate negative causal strengths,
(c) combine the positive and negative causal strengths, and
(d) derive conditional probabilities.

The CPT of node X has eight entries. This includes $P(\mathrm{X} \mid \mathrm{A, B, C})$, P(X | A, B, ¬C), . . . , and $P(\mathrm{X} \mid \neg\mathrm{A}, \neg\mathrm{B}, \neg\mathrm{C})$. These four steps are used to calculate each of these eight conditional probabilities. For instance, to calculate the probability $P(\mathrm{X} \mid \mathrm{A, B}, \neg\mathrm{C})$, the h values on the arcs connecting A and B to X and the g value on the arc connecting C to X are considered. Hence, the set of causal strengths is $\{0.9, -0.4, -0.5\}$.

2.1.1. *Aggregate the Positive Causal Strengths*

In this step, the set of causal strengths with positive influence are combined. They are aggregated using the equation

$$\mathrm{PI} = 1 - \prod_i (1 - C_i) \quad \forall C_i > 0$$

where C_i is the corresponding g or h value having positive influence and PI is the combined positive causal strength. For our example

$$\mathrm{PI} = 1 - (1 - 0.9) = 0.9$$

2.1.2. *Aggregate the Negative Causal Strengths*

In this step, the causal strengths with negative values are combined. The equation used for aggregation is

$$\mathrm{NI} = 1 - \prod_i (1 - C_i) \quad \forall C_i < 0$$

where C_i is the corresponding g or h value having negative influence and NI is the combined negative causal strength. Using this equation, the aggregate negative influence is found to be

$$\mathrm{NI} = 1 - (1 - 0.4)(1 - 0.5) = 0.7$$

2.1.3. *Combine Positive and Negative Causal Strengths*

In this step, aggregated positive and negative influences are combined to obtain an overall net influence. Mathematically,

$$\text{If PI} > \text{NI}$$

$$\text{AI} = \frac{\text{PI} - \text{NI}}{1 - \text{NI}}$$

$$\text{If NI} > \text{PI}$$

$$\text{AI} = \frac{\text{NI} - \text{PI}}{1 - \text{PI}}$$

Thus, the overall influence for the current example is

$$\text{AI} = \frac{(0.9 - 0.7)}{(1 - 0.7)} = 0.66$$

2.1.4. *Derive Conditional Probabilities*

In the final step, the overall influence is used to compute the conditional probability value of a child for the given combination of parents.

$$\begin{aligned} P(\text{child}|\,j\text{th state of parent states}) &= \text{baseline} + (1 - \text{baseline}) \times \text{AI, when PI} \geq \text{NI} \\ &= \text{baseline} - \text{baseline} \times \text{AI, when PI} < \text{NI} \end{aligned}$$

Using this equation, $P(\text{X} \mid \text{A, B}, \neg\text{C})$ is obtained as

$$P(\text{X}|\text{A, B}, \neg\text{C}) = 0.5 + 0.5 \times 0.66 = 0.863$$

These steps are repeated for the remaining seven entries of node Xs CPT. It should be noted that if the experts had sufficient time and knowledge of the influences, then they could have directly provided the CPT for each node in the IN instead of providing g and h values. Furthermore, after estimating the CPTs if some entries do not satisfy the experts, then those entries can be modified.

2.2. Dynamic Influence Nets

Influence nets were originally designed to capture static situations. Later, enhancements were made by Wagenhals et al.[31] and Haider and Levis[23] that enable INs to capture information and communication time delays present in a problem domain. The new constructs also allow a system analyst to associate time stamps with the actionable items. These time stamps describe the time instants at which actions are taken. Furthermore, the new enhancements also added the concept of time-varying (nonstationary) influences and self-loop. This section briefly discusses the important concepts related to DINs. Readers interested in a detailed and comprehensive discussion on DINs should refer to Haider and Levis.[23]

The modeling of the static causal relationships in a DIN is accomplished by creating a series of cause and effect relationships among variables representing set of actionable events and variables representing desired effect(s). The actionable events are drawn as root nodes (nodes without incoming edges), whereas the desired effect is modeled as a leaf node (node without outgoing edges). Typically, the root nodes are drawn as rectangles, whereas the nonroot nodes are drawn as rounded rectangles. The directed edge with an arrowhead between two nodes shows the parent node promoting the chances of a child node being true, whereas the roundhead edge shows the parent node inhibiting the chances of a child node being true. The values on the arc represent time delays, which correspond to communication delays, whereas the (nonmandatory) values on the nodes represent information processing time delays. The text associated with an actionable event represents the time at which the action is executed.

Figure 2 shows an example of a DIN. Nodes A and B represent the actionable events (root nodes), whereas node D represents the objective node (leaf node). The time delays associated with the arcs represent communication delays, that is, how long does it take before a belief change at a parent node influence the corresponding child node. For instance, the time delay on the link between A and C is 2, which says that it takes 2 time units before the influence of node A reaches node C. The text associated with the root nodes specifies the selected COA or input scenario. For instance, the text associated with node B says that initially the probability of B being true is 0.1. It reaches 0.6 at time 1 and the action is finally taken at time 5 (probability of 1). The text associated with node A is read in a similar manner.

The self-loop associated with nodes C and D models the dependence of these events on their previous states. A self-loop can also be used to model *decay* in the belief of a node when it receives no new information from its parents. The text associated with arcs represents time-varying influences. In almost every real-world situation, events happen and they influence other relevant events. As the time passes

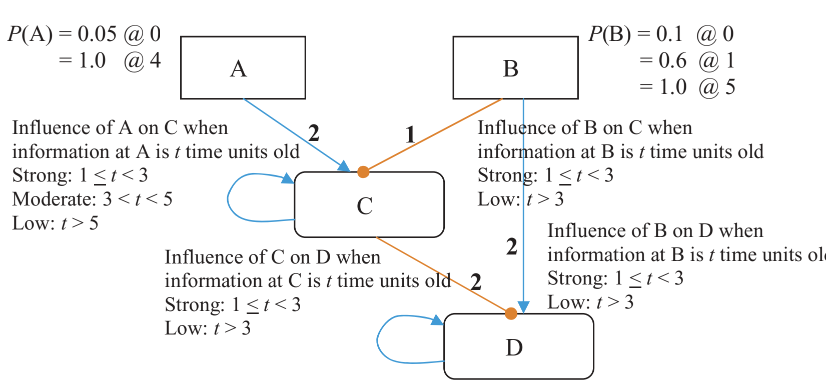

Figure 2. A dynamic influence net.

by, however, the influences lose their intensity, that is, the influences decay over time. Thus, an event having a very strong influence at the time of its occurrence on another event might have an insignificant influence after a certain period of time. Such influences are referred to as *time variant*. A DIN allows a system modeler to specify such time-varying or nonstationary influences. Instead of providing single-valued influences (as discussed in Section 2.1), a DIN allows the modeler to specify various strengths of influences and their corresponding windows of effectiveness. For instance, the text associated with the arc connecting A to C is read in the following manner: While computing the probability of C, if the information at A is less than 3 time units old, then A has a strong positive influence on C; if the information at A is 3 or 4 time units old, then A has a moderate positive influence on C; and finally if the information at A is more than 5 time units old, then A has a low positive influence on C. Similarly, B has a strong negative influence on C when the change that occurred at B is 1–2 time units old, whereas it has a low influence when the change occurred at B is more than 2 time units old. For simplicity, "strong influence" is assumed to mean that both h and g have the same values (0.9), though with opposite signs (one is positive and the other is negative depending upon the nature of arc). "Moderate influence" is mapped to the CAST logic values of 0.66 (and -0.66) and "low influence" is mapped to 0.33 (and -0.33).

Once a DIN is completely specified, the next step is to compute the probabilities of all nonroot nodes at different point in time. For the sample DIN of Figure 3, this includes computing probabilities of nodes C and D at different time instants. For

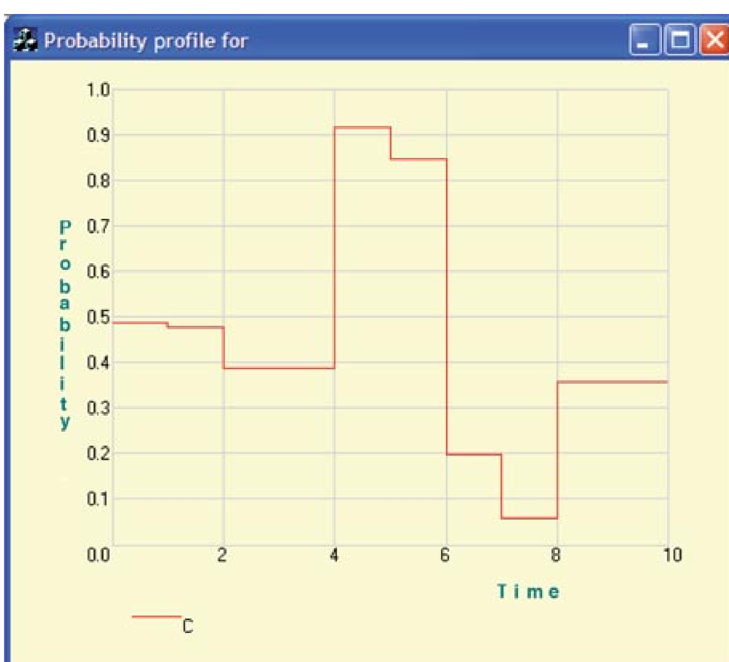

Figure 3. Probability profile of node C.

Table I. Nonstationary conditional probability tables for node C.

Parents combination	Time 2	4	6
$P(C_t \mid \neg A, \neg B, \neg C_{t-1})$	0.889	0.050	0.781
$P(C_t \mid \neg A, \neg B, C_{t-1})$	0.950	0.111	0.901
$P(C_t \mid \neg A, B, \neg C_{t-1})$	0.022	0.022	0.011
$P(C_t \mid \neg A, B, C_{t-1})$	0.050	0.050	0.025
$P(C_t \mid A, \neg B, \neg C_{t-1})$	0.950	0.950	0.975
$P(C_t \mid A, \neg B, C_{t-1})$	0.978	0.978	0.989
$P(C_t \mid A, B, \neg C_{t-1})$	0.050	0.889	0.099
$P(C_t \mid A, B, C_{t-1})$	0.111	0.950	0.219

instance, because of the provided input scenario, C receives updates from its parents (A and B) at times 2, 4, and 6. The probability of C is updated at time 2 because the probability of action B is changed at time 1 and since the time delay between B and C is 1, C receives the update at time 2. The other changes can be read in a similar manner (A having a new probability at 2 and B having a different probability at time 5). Each update in the probability of C requires computing new (nonstationary) CPTs with the help of time-varying CAST logic parameters. The process is similar to the one shown in Section 2.1, with the exception that this time around the influences are not static, that is, for each time instant, a different set of g and h is used depending upon the strength (strong/moderate/low) of the corresponding influence. The nonstationary CPTs for node C are shown in Table I. It should be noted that the nonstationary CPTs are computed for each instant of time and not just for the time instants shown in Table I. The simple reason is that even without receiving a new update from any of its parent, some of the existing influences may lose their strengths, which results in the computation of a new CPT.

The purpose of building a DIN is to evaluate and compare the performance of alternative courses of action. The impact of a selected COA on the desired effect is analyzed with the help of a *probability profile*. A probability profile draws the probabilities of a node against the corresponding time line. The probability profile of event C is shown in Figure 2.

Formally, the following items characterize a DIN:

(1) The nodes of a DIN are set of random variables. All the variables in the DIN have binary states.
(2) A set of directed links that connect pairs of nodes. A node can also have an optional self-loop.
(3) A pair (c, t) for each link, where c is a list of tuples representing the CAST logic parameters. For each element in c, a corresponding time interval is defined in t. This interval represents the time during which the corresponding element in c is in effect. In general, (c, t) is defined as $([(h1, g1)\ (h2, g2), \ldots, (hn, gn)], [(t11, t12), (t21, t22), \ldots, (tn1, tn2)]$, where $ti1 < ti2$ and $tij > 0$,$i = 1, 2, \ldots, n$, and $j = 1, 2$.
(4) Each nonroot node has an associated baseline probability, whereas a prior probability is associated with each root node.
(5) Each link has a corresponding delay d (where $d \geq 0$) that represents the communication delay.

(6) Each node has a corresponding delay e (where $e \geq 0$) that represents the information processing delay.
(7) A pair (p, t) for each root node, where p is a list of real numbers representing probability values. For each probability value, a corresponding time interval is defined in t. In general, (p, t) is defined as $([p1, p2, \ldots, pn], [[t11, t12], [t21, t22], \ldots, [tn1, tn2]]$, where $ti1 < ti2$ and $tij > 0$, $i = 1, 2, \ldots, n$, and $j = 1, 2$.

The last item in this list is referred to as an input scenario or a COA.

3. DYNAMIC BAYESIAN NETWORK

Dynamic Bayesian networks are an extension of BNs designed to model dynamic systems evolving over time. They subsume an important family of dynamic models commonly known as hidden Markov models. Dynamic Bayesian network have been successfully applied in many areas including, but not limited to, speech recognition, gene sequencing, motion detection, etc.

A DBN works by discretizing time and creating instances of variables in a BN for each point in the time interval under consideration. The process starts with the identification of static cause and effect relationships among the variables and then by repeating (unrolling) the same structure for a given number of time slices. Links are drawn between variables having temporal dependencies. The resultant unrolled network is treated as a static BN and is solved using standard inference algorithms (either exact or approximate).

Figure 4 shows a DBN unrolled for four time slices. Each slice consists of a fixed network structure that models the static cause and effect relationships among the system variables. The temporal links connecting structures in different time slices capture system's dynamics. Along with other nontemporal links, they define the conditional distribution of variables, that is, how the state of a variable at time t depends on the state of other variables in the same time slices and the preceding time slices. If the temporal links connect variables in time slice t to only variables in the immediate past (time slice $t - 1$) then the DBN is first-order Markovian.[32] Although

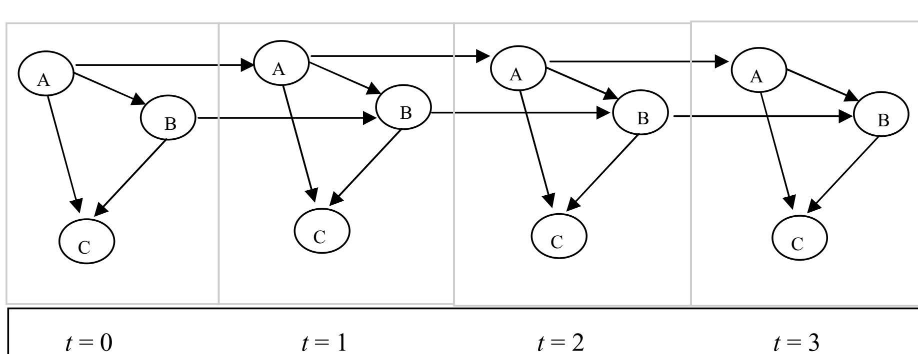

Figure 4. A sample dynamic Bayesian network.

a DBN can be any order Markovian, for the simplicity, DBNs are built as first-order Markovian. One more assumption that simplifies the specification of a DBN is that the changes in the state of variables are caused by *stationary processes*, that is, the transition probability distributions are invariant between time steps. Mathematically, this implies

$$P(x_t|pa(x_t)) = P(x_{t-1}|pa(x_{t-1})), \qquad where \;\; t = 1, \ldots, n$$

4. TRANSFORMATION ALGORITHM

This section presents a transformation algorithm that converts a DIN into an equivalent DBN. The algorithm is an extension of the work by Haider and Zaidi[30] to transform a TIN into a time-sliced BN. The algorithm synergizes the modeling capabilities of both probabilistic modeling paradigms, that is, DIN and DBN. It provides a system analyst the capability to use a DIN for capturing the uncertainties present in a complex situation with minimum amount of information and for selecting an effective COA that maximizes the likelihood of achieving the desired effect During the execution of the COA, if information (evidence) about certain events is received, then the DIN is transformed into an equivalent DBN. Beliefs of other nodes are then updated using standard DBN inference algorithms and, if required, the selected COA can be revised. The actions already taken by that time instants are considered hard evidence and are incorporated as constraints during the strategy revision process. The important steps of the algorithm are highlighted in Table II, whereas the algorithm is explained with the DIN in Figure 3 and DBN in Figure 5.

The transformation algorithm first determines the maximum path length that exists between the root nodes and the target node. For the DIN of Figure 3, M is 4, which is the length of path A–C–D. The algorithm then identifies the maximum time stamp from all the time stamps associated with actionable events. The maximum time stamp in Figure 3 is 5, which is associated with node B. The algorithm then draws 9 ($M = 4$, $S = 5$) time slices. The connections between nodes are established according to the time delays associated with the corresponding arcs. For instance, the arc delay between B and D is 2, thus D5 is connected to B3, D4 is connected to B2, and so on. Similarly, the arc delay between B and C is 1, thus C5 is connected to B4, C4 is connected to B3, and so on. The subsequent indices of a root node

Table II. The transformation algorithm.

1. Let M be the maximum path length between the root nodes and target nodes.
2. Let S be the maximum time stamp associated with the root nodes as provided by the input scenario.
3. Draw $M + S$ time slices in the resultant dynamic Bayesian network. The nodes are connected according to the time delays on the arcs. The subsequent indices of a root node (representing actionable events) are also connected except for the time the probability of an action is explicitly defined.
4. The nonstationary conditional probability tables are computed using the time-varying CAST logic parameters.
5. The input scenario is assigned as prior probabilities of actionable events.

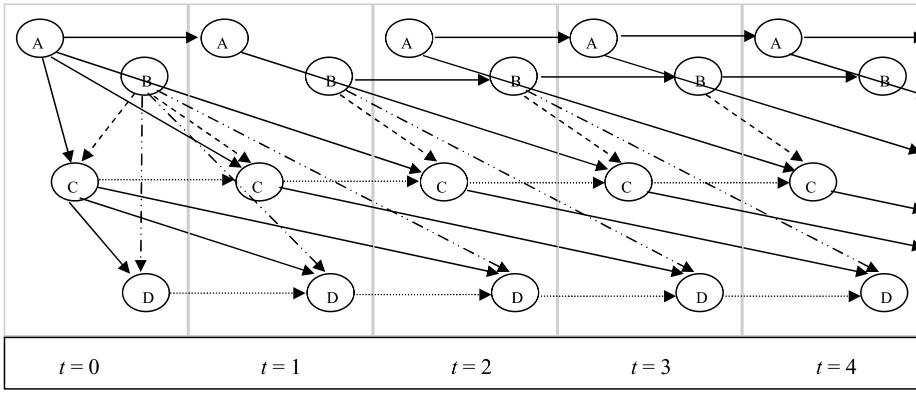

Figure 5. The transformed dynamic Bayesian network of dynamic influence net of Figure 3.

(representing actionable events) are also connected except for the time when an action is taken. For instance A1 is not connected to A2 because the probability of A is changed at time 2. Similarly, B0 is not connected to B1 because the probability of B is changed at time 1. Figure 5 shows the resultant unrolled DBN for first five time slices. It should be noted that the different line styles (solid, dashed, etc.) in Figure 5 have no semantic meanings attached to them. They are simply drawn in different styles to make them more readable.

Once the structure of DBN is constructed, the next task is to assign parameters to it. This step involves computing nonstationary CPTs according to the procedure described in the previous section and demonstrated in Table I. Moreover, the input scenarios are assigned as prior probabilities. For instance, the prior probability of A0 is set to 0.05, whereas the prior probability of A2 is 1. Similarly, the prior probability of B0, B1, and B5 is set to 0.1, 0.6, and 1.0, respectively. The assignment of prior probabilities and nonstationary CPTs completes the transform algorithm and produces a fully specified DBN.

As stated earlier, the main advantage of DINs lies in their ability to immensely ease the knowledge acquisition process by asking only linear number of parameters and having the provision to model positive and negative influences. Once built, they are used to identify the best COA in a particular problem domain. Haider and Levis[28,29] have developed evolutionary algorithms and particle swarm optimization–based techniques to automate this best COA identification process. The process includes the selection of actions that need to be taken and the timings of their execution. These computational intelligence–based approaches can also handle certain types of causal and temporal constraints that can exist among actionable events. It should be noted, however, that besides nodes representing actionable events, a DIN also has nodes that represent uncertain events. They are called uncertain because not enough information is available about these events at the start of a campaign. In many situations, it is often the case that the states of these uncertain events become known with certainty during the execution of the COA. This new information may either be favorable to the mission's objective or be unfavorable to it. In both cases, it is desirable to revise the current strategy after incorporating (a) this additional information and (b) the set of actions taken so far in the campaign. The process is called belief updating. Currently, DINs do not have the capability to update beliefs in the light of newly arrived information. The transformation algorithm presented in this paper aims to overcome this limitation. After transforming the DIN into an equivalent DBN, new evidence is easily incorporated and beliefs of other nodes are updated using standard belief-updating algorithms. For instance, if evidence is received about event D at time 5, then all the subsequent indices of D in the corresponding DBN are also treated as evidence nodes unless the duration of evidence is for a single or few time slices. If the time associated with the new information is greater than the number of slices already drawn in the DBN, then more slices are added to it. Once evidence is incorporated, the computational intelligence–based techniques, discussed above, are run again on the updated DBN to determine whether the current strategy needs revision. The actions already taken before the arrival of evidence are provided as constraints to the automated COA selection process. Because of the scope and space limitation, the complete process of automated COA selection is not presented in this paper. Readers interested in a detailed understanding of the process should refer to Haider and Levis.[28,29]

5. CONCLUSIONS

The paper presented an algorithm to transform a DIN into a DBN. The transformation aims to combine the capabilities of both modeling paradigms. Dynamic Bayesian networks comprise an immensely powerful tool in representing highly complex situations requiring strategic-level decision making in a compact and easy-to-understand manner. Moreover, they require only linear number of parameters to represent nonstationary processes. Once completely specified, a DIN is used to analyze the performance of difference courses of action in terms of their chances of achieving the desired effect(s). During the execution of a COA, however, if new information about particular uncertain events becomes available, then the DIN does not have a mechanism to incorporate this information. On the other hand, several belief-updating algorithms are available for DBNs. However, DBNs themselves suffer from the intractability of knowledge acquisition. Thus, the presented methodology combines the advantages of both modeling paradigms. The methodology suggests using DIN as a front-end tool for modeling complex situations and analyzing COAs. During the execution of the selected COA, if evidence about certain event is received, then the DIN is transformed into a DBN and the beliefs of other events in the network are revised using standard belief-updating algorithms. If required, the COA itself can be revised by treating the actions taken so far as constraints.

References

1. Mittal A, Kassim A. Bayesian network technologies: Applications and graphical models. Hershey, PA: IGI Publishing; 2007.
2. Pourret O. Naim P, Marcot B. Bayesian networks: A practical guide to applications. New York: Wiley; 2008.

3. Pearl J. Probabilistic reasoning in intelligent systems: Network of plausible inference. San Mateo, CA: Morgan Kaufmann; 1987.
4. Murphy K. Dynamic Bayesian networks: Representation, inference and learning, PhD Thesis, University of California, Berkley, 2002.
5. Boyen X, Koller D. Tractable inference for complex stochastic processes. In: Proc Conf Uncertainty in Artificial Intelligence, Madison, WI, 1998. pp 33–42.
6. Doucet A, de Freitas N, Murphy K, Russell S. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In: Proc 16th Conf Uncertainty in Artificial Intelligence, Stanford, CA, 2000. pp 176–183.
7. Kjaerulff U. A computational scheme for reasoning in dynamic probabilistic networks. In: Proc 8th Conf Uncertainty in Artificial Intelligence, Stanford, CA, 1992. pp 121–129.
8. Murphy K, Weiss Y. The factored frontier algorithm for approximate inference in DBNs. In: Proc 17th Conf Uncertainty in Artificial Intelligence, Seattle, WA, 2001. pp 378–385.
9. Takikawa M, d'Ambrosio B, Wright E. Real-time inference with large-scale temporal Bayes nets. In: Proc 18th Conf Uncertainty in Artificial Intelligence, Alberta, Canada, 2002. pp 477–484.
10. Burns B, Morrison CT. Temporal abstraction in Bayesian networks, In: AAAI Spring Symposium, Palo Alto, CA, 2003.
11. Figueroa GA, Sucar LE. A temporal Bayesian network for diagnosis and prediction. In: Proc 15th Conf Uncertainty in Artificial Intelligence, Stockholm, Sweden, 1999. pp 13–20.
12. Galan SF, Diez FJ. Networks of probabilistic events in discrete time. Int J Approx Reas 2002;30:181–202.
13. Galan SF., Figueroa GA. Diez FJ, Sucar LE. Comparative evaluation of temporal nodes Bayesian networks and networks of probabilistic events in discrete time. In: Third Mexican International Conference on Artificial Intelligence: MICAI 2004, Mexico City, Mexico, April 26–30, 2004. pp 498–507.
14. Hanks S, Madigan D, Gavrin J. Probabilistic temporal reasoning with endogenous change. In: Proc 11th Conf Uncertainty in Artificial Intelligence, Montreal, Canada, 1995. pp 225–245.
15. Santos E Jr, Young JD. Probabilistic temporal network: a unified framework for reasoning with time and uncertainty. Int J Approx Reas 1999;2:263–291.
16. Tawfik AY, Neufeld E. Temporal Bayesian networks. In: Proc 1st Int Workshop on Temporal Representation and Reasoning (TIME), Pensacola, FL, May 1994. pp 85–92.
17. Chang KC, Lehner PE, Levis AH, Zaidi SAK, Zhao X. On causal influence logic. Technical Report, George Mason University, Center of Excellence for C3I, Fairfax, VA, 1994.
18. Rosen JA, Smith WL. Influence net modeling with causal strengths: an evolutionary approach. In: Proc Command and Control Research and Technology Symposium, Naval Post Graduate School, Monterey, CA, June 1996.
19. Agosta JM. Conditional inter-causally independent node distributions, a property of noisy-OR models. In: Proc 7th Conf Uncertainty in Artificial Intelligence, Los Angeles, CA, 1991. pp 9–16.
20. Drudzel MJ, Henrion M. Intercausal reasoning with uninstantiated ancestor nodes. In: Proc 9th Conf Uncertainty in Artificial Intelligence, Washington, DC, 1993. pp 317–325.
21. Murphy K. Weiss Y, Jordan M. Loopy-belief propagation for approximate inference: an empirical study. In: Proc 15th Conf Uncertainty in Artificial Intelligence, Stockholm, Sweden, 1999. pp 467–475.
22. Wagenhals LW, Wentz LK. New effects-based operations models in war games. In: Proc 8th Int Command and Control Research and Technology Symposium, Washington, DC, June 2003.
23. Haider S, Levis AH. Modeling time-varying uncertain situations using dynamic influence nets. Int J Approx Reas 2008;49(2):488–502.
24. Wentz LK, Wagenhals LW. Effects based operations for transnational terrorist organizations: assessing alternative courses of action to mitigate terrorist threats. In: Proc 9th Int Command and Control Research and Technology Symposium, San Diego, CA, June 2004.

25. Wagenhals LW, Janssen RA, DeGregorio EA. Model interoperation for effects based planning. In: Proc 9th Int Command and Control Research and Technology Symposium, San Diego, CA, June 2004.
26. DeGregorio EA, Janssen RA, Wagenhals LW, Messier RH. Integrating effects-based and attrition-based modeling. In: Proc 9th International Command and Control Research and Technology Symposium, San Diego, CA, June 2004.
27. Wagenhals LW, Levis AH. Course of action analysis in a cultural landscape using influence nets. In: Proc IEEE Symposium on Computational Intelligence for Security and Defense Applications, Honolulu, HI, April 2007.
28. Haider S, Levis AH. Effective courses-of-action determination to achieve desired effects. IEEE Trans Syst Man Cybern Part A: Syst Hum 2007;37(6):1140–1150.
29. Haider S, Levis AH. Finding effective courses of action using particle swarm optimization. In: Proc World Congress on Computational Intelligence, Hong Kong, 2008. pp 1135–1140.
30. Haider S, Zaidi AK. Transforming timed influence nets into time sliced Bayesian networks. In: Proc Command and Control Research and Technology Symposium, San Diego, CA, 2004.
31. Wagenhals LW, Shin I, Levis AH. Creating executable models of influence nets with coloured Petri nets. Int J Softw Tools Technol Transfer 1998;2(2):168–181.
32. Kjaerulff U, Madsen AL. Bayesian networks and influence diagrams: A guide to construction and analysis. New York: Springer; 2007.
33. Wagenhals LW, Reid TJ, Smillie RJ, Levis AH. Course of action analysis for coalition operations. In: Proc 6th Int Command and Control Research and Technology Symposium, Annapolis, MD, June 2001.