

CONVERTING INFLUENCE NETS WITH TIMING INFORMATION TO A DISCRETE EVENT SYSTEM MODEL, A COLORED PETRI NET*

LEE W. WAGENHALS AND ALEXANDER H. LEVIS

System Architectures Laboratory
School of Information technology and Engineering
George Mason University
Fairfax, VA 22030-4444, USA
{lwagenha, alevis}@gmu.edu

ABSTRACT

Recent research has shown how to incorporate time in probabilistic modeling techniques called influence nets that are used to model complex political/military situations. By adding timing information to these models, which are static equilibrium models, they are converted to discrete event system models that can be represented as colored Petri nets. Executing these CP nets reveals the dynamic changes in the probability values of key events that are modeled as propositions in the influence net. This paper illustrates how the state space analysis capability of Design/CPN was used to verify the behavior of a generic CP net model generated from the influence net. First, the implications of incorporating time in an influence net are presented along with the type of behavior that the discrete event model should have. A procedure for interconnecting CP net modules to create the overall model is presented. Finally, the state space analysis capabilities of Design/CPN are used to verify the behavior of the model and reveal interesting properties of the dynamical model that are not intuitively obvious from the structure of the model.

1 INTRODUCTION

Influence nets [11] are inspired by and are similar in form to Bayesian nets [2, 3, 10]. Because they are simpler to construct and computationally less burdensome than the Bayesian net, they can support the development of models of situations by a group of subject matter experts who need not be experienced in Bayesian nets. The fundamental assumption in influence nets, that the parents of any node are independent, causes them to behave in a similar, but not identical, manner to Bayesian nets of the same form. This assumption also facilitates the introduction of timing information which is a fundamental enabler of this research.

The influence net models are created to identify a set of controllable events that collectively have the maximum positive influence on the objectives modeled in the network. These events are called actionable events. Analysts create the influence net model of a situation using three types of nodes (hypotheses). Terminal nodes represent the effects or objectives that are desired as the result of actions to be taken. They have no outgoing arcs. Input nodes model the occurrence of the actionable events that may directly or indirectly influence or cause the objectives to occur. The third type of node is the intermediate node. These nodes model propositions that provide influencing links between the actionable events and the objectives. Once the model is constructed, a sensitivity analysis is performed to identify the actions that have the most impact on the objectives. These actions represent the un-sequenced and un-timed elements of a *Course of Action* (COA) which is defined as a timed sequence of actionable events.

In current practice, once the un-sequenced COA has been determined, the set of selected actions are provided to operational planners. They evaluate the availability of the resources needed to cause

* This work was supported in part by the Air Force Office of Scientific Research under Grant No. F49620-98-1-0179

the occurrence of the actionable events and determine how to schedule those resources to achieve those actions. After the plan has been formulated and approved, it is executed under the direction of operational controllers, who make adjustment in the tasking of resources as the plan unfolds.

Several observations about the current process are in order:

1. Bayesian net or influence net are static equilibrium models. They are not dynamic; there is no concept of time. They yield a joint probability distribution that is consistent with any knowledge about the hypotheses in the net. In evaluating situations, they provide the marginal probability of the nodes that represent the objective hypotheses given occurrence of a set of input actions.
2. The probabilistic model does not provide information about the impact of sequencing or timing of the actionable events on the objectives.
3. There is limited collaboration between the situation analyst, the operational planners, and the operational controllers who design and implement the COA. Each receives an input from the other, but little dialog, discussion, or debate takes place between the actors.

1.1 CONTRIBUTION

This research provides a formal procedure for converting the information contained in an influence net along with timing information into an executable model of the situation that will generate the timed sequence of probability values for all nodes in an influence net for a given timed sequence of actionable events that defines the COA. This procedure clearly shows the time phased impact of the scheduling of resource activities on the objectives that are desired from the set of actions. A software application, written in CPN/ML, that automatically generates the CP net equivalent of the influence and provides the charting capability to display the impact of any COA on the objectives has been documented [12]. Furthermore, the researches have used the state space analysis tools of Design/CPN [4, 5, 6, 7] to verify the behavior of the CP net generated by the code.

The remainder of this paper is organized as follows. The next section shows how adding timing information changes the influence net model to a discrete event system that can be modeled using a CP net. Section 3 reviews the CP nets that are generated by the CPN/ML code and describes how they operate to generate probability profiles. Section 4 describes the state space analysis that was done to verify the behavior of the CP nets. The last section provides future directions for research.

2 INFLUENCE NET AS A DISCRETE EVENT SYSTEM

The influence net is based on causal relationships between propositions. In constructing influence nets, a temporal precedence is assumed between a causing proposition and the propositions that it influences. This assumption is consistent with most treatments of models of causality [10]. This in turn implies a sequencing of the probability of the propositions in the model as the propositions of the initial nodes become true and their effects propagate through the network. We assume that the cause and effect relationship between nodes is based on some real world phenomenon. This can be expressed as "Proposition A can trigger Proposition B (with some probability) after a time delay, d ." This time delay is caused by a real world phenomenon, such as a communication process, that transfers the knowledge of proposition A to an underlying entity that can cause Proposition B to occur. We further assume that it is possible to calculate or estimate the delay.

This description suggests that the influence net is an abstraction of an underlying dynamical system that consists of concurrent and asynchronous processes. Associating time with the arcs of the influence net introduces more complex behavior to the model of the system. This behavior can be modeled by converting the influence net to a discrete dynamical model. Any model so created must have at least three characteristics,

It must be capable of computing the marginal probability of any node, given a change in the state of one or more of its parents.

It must be capable of modeling time.

It must preserve the locality principle of the causal theory.

While several modeling frameworks could be considered, Petri nets seem most appropriate. Because of the need to compute the marginal probabilities, a Colored Petri net is required.

Because, in its intended use, the influence net is a model of causal influences where inputs will be provided only to nodes with no parents. We assume a sub-set of these input nodes will represent the actionable events, propositions that we have the ability to cause with certainty. Thus, it is assumed that the probability values of these nodes will either be zero or one. The initial probability value of all nodes representing actionable events is assumed to be zero. We assume each actionable event will be made to occur at some time, thus each such node will change from a probability value of zero to a value of one. Whenever such a node changes from zero to one or F to T, its effect will propagate through the network to one or more terminal nodes.

From this perspective, we can re-cast the model of the influence net into a discrete event system model as follows:

- Let M be the set of nodes in the influence net, $m_i \in M$
- Partition the set of nodes of the influence net as follows:
 - Let I be the set of input nodes representing the actionable events.
 - These are the nodes without parents
 - Let N be the set of non-input nodes; $I+N=M$
- Define the state of a node m_i of the influence net be its probability, $P(m_i)$
 - Let \mathbf{u} be the input to the model represented by the set of the nodes in I ; a vector of probability values from the elements of I
 - Let \mathbf{x} be the state vector of the set of nodes in N
 - Let \mathbf{y} be a vector of probability values of a subset of the nodes in N , these are the nodes of interest, usually the objective nodes, that are the output of the model
 - Let k be variable whose domain is the set of integers that is used as an index to indicate a particular value of \mathbf{u} , \mathbf{x} , or \mathbf{y} in a sequence of values

An input episode will be defined as a sequence of changes in the values of the elements of \mathbf{u}_k . In the initial state, none of the actionable events has occurred so $\mathbf{u}_k = 0$. As each actionable event occurs, the corresponding value of the element of \mathbf{u}_k changes from zero to one. Each element changes value once. The final state of the input vector is the unit vector, $\mathbf{u}_k = 1$.

With these definitions, it is possible to view an influence net as a discrete event system under the following conditions. (1) The state of the system is the vector of marginal probability values of each of the nodes in the N . (2) There is a set of admissible inputs for the nodes with no parents. As a discrete event system, there is a single initial state defined as the equilibrium probability values, of all the nodes in N , that would be calculated by the influence net with the probability value of all of the input nodes set to zero. Furthermore, there is a single final state, regardless of the sequence of the actionable events, that is the set of probability values computed by the static influence net model with all input nodes set to one. We can consider the input space, U , and the output, space, Y , as hypercubes. Each allowable input sequence can be represented as discrete changes of position in the input hypercube. Each input sequence induces a finite sequence of changes in the output space, starting with the single initial point and ending at the single final output point. Each sequence of changes represents a probability profile over the set of nodes in the output space. Figure 1 illustrates these concepts.

Figure 1 is based on a model with three input and three output nodes or variables. In the input space, all input sequences start at the origin of the input space. For this initial point in the input space, there is a corresponding point, marked P_{00} in the output space. Each allowable input sequence can be viewed as a set of discrete jumps from the origin of the input space to various "corners" of the hypercube, ending at the point $[1, 1, \dots, 1]$. Each sequence induces a piece-wise set of jumps or steps through the output space, culminating at a single final point, labeled P_f in the figure. Two input sequences are highlighted, input sequence 1 is shaded, and input sequence 2 is solid. The corresponding notional paths through the output space are also shown. The projection of the paths in the output space on the various axes provides the probability profile for each output variable.

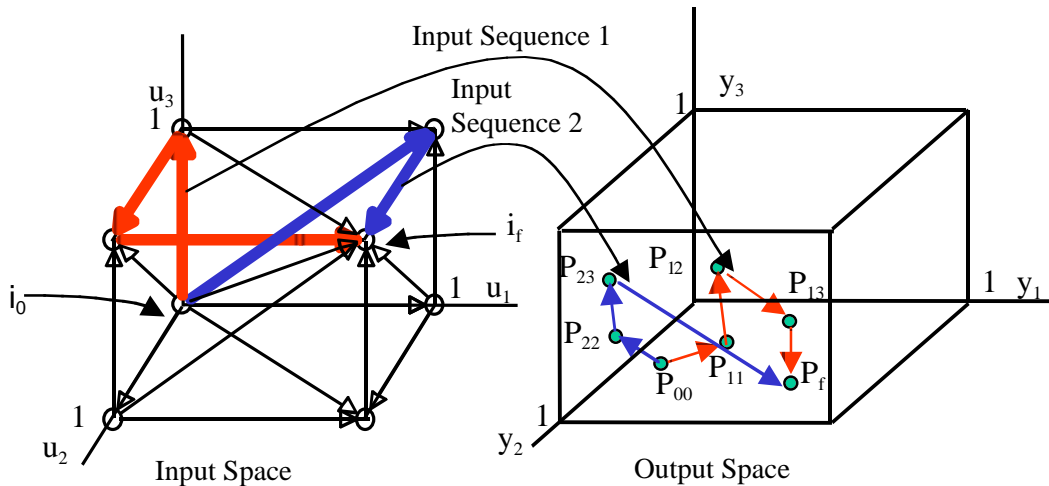


Figure 1 Admissible Input Sequences Induce Output Sequences

It is important to recognize, that there is no time or clock associated with the sequences in the above description. As described in detail by Cassandras [1], two classes of models are available that capture the behavior of discrete event system, finite state automata and Petri nets, with Petri nets being the more general model.

So far the case has been made that an influence net can be modeled as an untimed discrete event system. We need to extend the approach to incorporate time, creating a timed discrete event system model. Figure 2 shows an example of a influence net with time delays associated with the influences. A time line annotated with events is shown below the net. A time stamp can be associated with each state change of the set of input nodes. It is assumed that updates of the marginal probability of any node occur immediately after the associated time delay.

In the time line of Figure 2, the actionable events, E and F both occur at time $t = 0$, that is, they have a time stamp of zero. As shown on the time line, at $t = 1$, node A is updated due to the change in state of node E. This update uses the $P[E] = 1.0$ and the probability of $P[F] = 0.0$ even though both are one. This is denoted as $U_{A|E}$ on the time line. At $t = 2$, node A updates its state again due to the change in node F that occurred at $t = 0$. At $t = 3$, two updates occur simultaneously. Node B is updated due to the change of node F, and node C updates due to the first update of node A. Finally, at $t = 4$, node C becomes aware of the new states of both nodes A and B and thus updates its state accordingly.

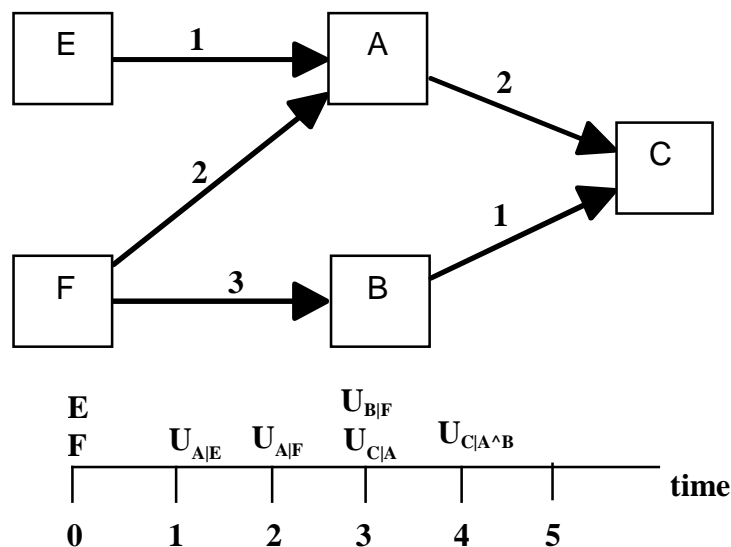


Figure 2 Timed Influence Net

This example illustrates how both the time delays associated with the influences and the time stamps of the inputs effect the order and the number of updates that occur in all of the nodes of the net. The order of the updates also affects the probability values for each update. By converting the influence net to a timed CP net, this dynamic behavior can be captured.

3 A CPN MODEL OF A TIMED INFLUENCE NET

To create a CPN model, first consider a general node in an influence net. Recall that each node in an influence net can be categorized as to one of three types, an initial node, and terminal node, or an intermediate node. The actionable events are always initial nodes, and the outcomes are the terminal nodes. Figure 3 presents a representation of an intermediate node in an influence net. Node A has n parents and m children. When the state of any parent changes, Node A will be ready to update its state after the time delay indicated on the appropriate incoming arc. Similarly, when Node A changes state, the evidence of that state change will be available at each of its children after the delay indicated on the appropriate departing arc. Note that an initial node is just an intermediate node with no parents, and a terminal node is an intermediate node with no children.

Figure 4 shows a generalized CP net design for intermediate Node A of Figure 3 with n inputs and m output arcs. The global declaration node (GDN) is shown in Figure 5. This first half of the GDN specifies the color sets and the variables used in the model. The second half of the GDN contains a set of functions that are used to compute the marginal probability of a node from a list of the marginal probabilities of the node's parents and a list containing the values of conditional probability distribution of the node. The function used in the CP net arc inscription is `comp marg(<list of parents>, <list of conditional probability values>)`.

The net is composed of $n + m + 1$ places and $1 + m$ transitions. The places named IN1 and IN n represent inputs to Node A from each of its parents. The tokens are of color set Marg that is a pair composed of a real and an integer. The real portion contains the marginal probability of the parent node, and the integer acts as a control for the transition `td`. This transition is enabled whenever there are tokens in all of the places representing the input nodes and the guard function evaluates to true. The subnet has an initial marking for all of the input places and each place is both in the preset and the post set of any transition it is connected to, so that any firing of such a transition removes and replaces the token. Thus there will always be a token in each place of color set Marg. Furthermore, the guard function of the `td` transition evaluates to true whenever one or more of the integer components of the input places is greater than zero. This means that whenever a parent is updated, it must replace the current token with a new token containing the new marginal probability value and the control integer set to one.

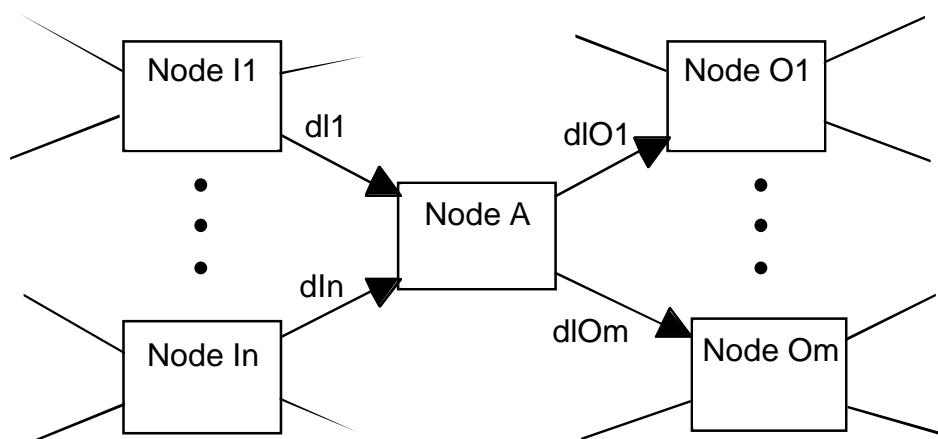


Figure 3 Generic Intermediate Node in an Influence Net

The `td` transition has a place of color set *Rule* in both the preset and the postset (a self loop). The *Rule* place contains tokens that are triples. The first element of the triple is an integer used as a counter. The second element of the triple contains a list of the values of the conditional probability distribution of the node. The third element is a list of the time delay values for all of the paths to the

children of the node. When *td* fires, it removes the token in the *Rule* place and replaces it with a similar token with the counter incremented by one.

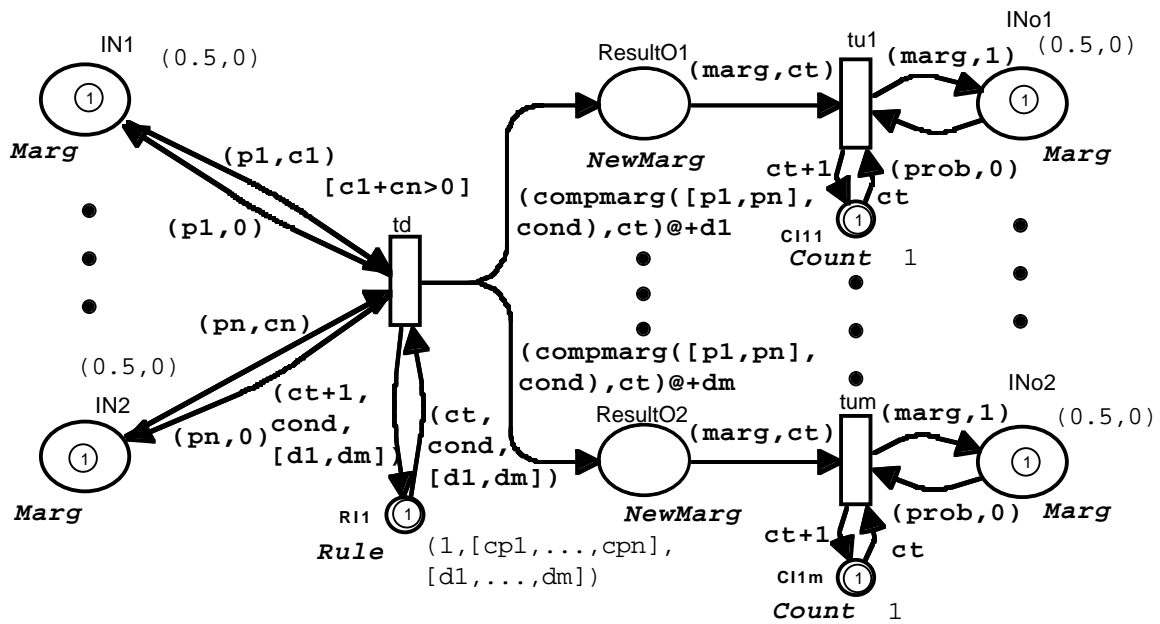


Figure 4 CP Net of an Intermediate Node of an Influence Net

```

color Control = int;
var c1,c2,c3,c4,c5,c6,c7,c8,c9,c10,c11,c12,cn: Control;
color Delay = int;
var d1, d2, d3, d4, d5, d6, d7, d8, d9, d10, d11, d12, dm: Delay;
color Prob = real;
var prob, marg ,p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12,pn: Prob;
color Marg = product Prob * Control;
color Count = int;
var ct: Count;
color CondList = list Prob;
var cond: CondList;
color DelayList = list Delay;
color Rule = product Count * CondList * DelayList ;
color NewMarg= product Prob * Count timed;

fun constlist 0 = []
  | constlist n = (n-1)::constlist (n-1);
fun twopower 0 = 1
  | twopower p = 2*(twopower (p-1));
fun detbase 1 n = [n]
  | detbase p n =
    let val q = (n div (twopower (p-1)))
    in q::(detbase (p-1) (n-(q*(twopower (p-1)))))) end;
fun probdet [] [] = 1.0
  | probdet (p::ps) (y::ys) = (if y=0 then p else (1.0-p))*(probdet ps ys);
fun multlist [] [] = 0.0
  | multlist ((x:real)::xs) ((y:real)::ys) = (x*y) + (multlist xs ys);
fun compmarg(pli,problast)=multlist (map (probdet pli) (map (detbase
(length(pli))) (constlist (length(problast)))))) problast;

```

Figure 5 Global Declaration Node

When transition td fires, it calculates a new marginal probability for Node A by reading all of the current values of the inputs. These are the arguments for the $comp\text{marg}$ function that is on each of the td transition's output arcs that are going to the child nodes of the influence net. The first argument is a list of the marginal probabilities of all of the parents $[p_1, \dots, p_n]$ and the second is a list of the condition probability values for Node A that is stored in the place of color set *Rule*. Transition td sends a token with the new marginal probability value, to m different output places of type *NewMarg*. *NewMarg* is a timed color set that is a double composed of a real, for the probability value, and an integer. Each of these tokens is given a time stamp consistent with the delay associated with the propagation of the influence between Node A and each of its children (d_1, d_2, \dots or d_m). After the appropriate delay, each transition tu_1 through tum fires and exchanges the token in the input place of the child with the new marginal probability of Node A and the control value set to one. This "signals" the child, which has a CP net that is similar to that for Node A, that a new update has arrived.

The purpose of the place of color set *Count* is to implement a first in, first out (FIFO) protocol on token in the *NewMarg* places. The FIFO protocol is required because it is possible, depending on the firing sequence, to have transition td fire more than once without advancing the simulation time clock. This means that more than one token can exist in the *NewMarg* places with the same time stamp. It is necessary to preserve the order of the generation of tokens with the same time stamp to ensure that the correct final state is reached.

The FIFO protocol is implemented by the second component of token in the *Rule* place which is a counter that is initialized to one and whose value is incremented each time transition td fires. The place with color set *Count* contains a similar counter that increments whenever its transition, tui , fires. If there are multiple tokens in the *NewMarg* place, they will be consumed in the order in which they were generated by transition td because the ct variable of the token must match the value of the ct in the *Count* place.

CP nets representing input and terminal nodes are similar to an intermediate node as shown in Figures 6 and 7. Note that the input node has a single input place of timed color set *NewMarg* to allow for the time stamps to be incorporated on the input tokens that represent the occurrence of an actionable event. The terminal node has a single output to which there is no delay associated. The *NewMarg* color set is used as the output place named *ResultO1*. It contains an initial marking that is the marking of the node that results if all the inputs are set to zero. The same *Rule* place is used to provide a counter for the tokens generated in the *ResultO1* place indicating the sequence in which they were generated.

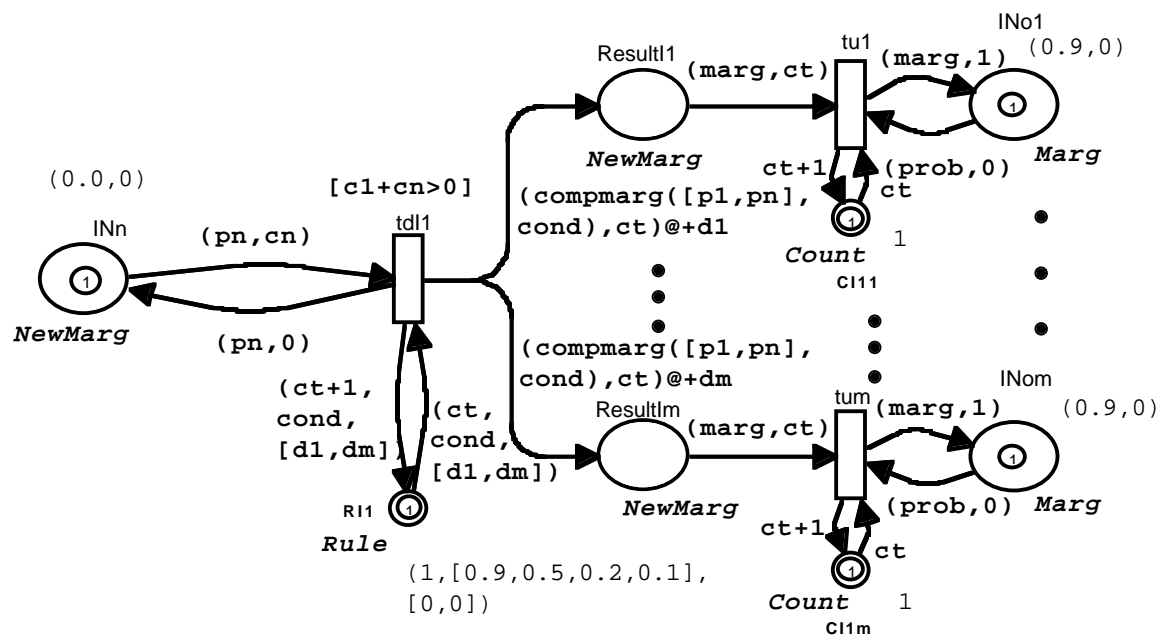


Figure 6 Generic CP Net of an Input Node of an Influence Net

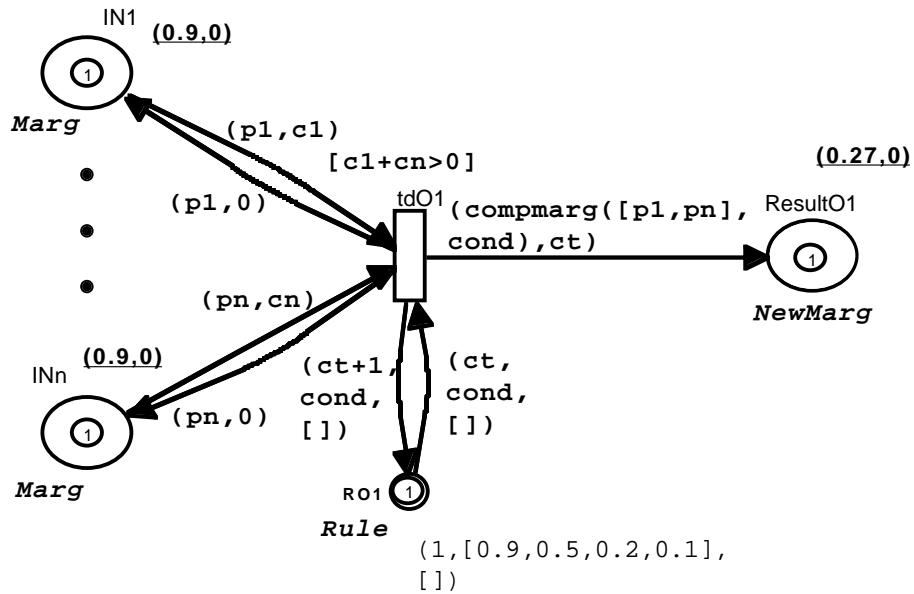


Figure 7 Generic Output Node

Converting an influence net to a time colored Petri net is straight forward. Indeed, an automatic routine for doing this has been written in CPN/ML [12]. To illustrate how the sub nets are interconnected, consider the three node timed influence net of Figure 8 with time delays d_1 , d_2 , and d_3 . Node A is a input or source node, node B is an intermediate node, and node C is a terminal or sink node. The corresponding colored Petri net for this influence net is shown in Figure 9. Most of the arc inscriptions have been suppressed for clarity.

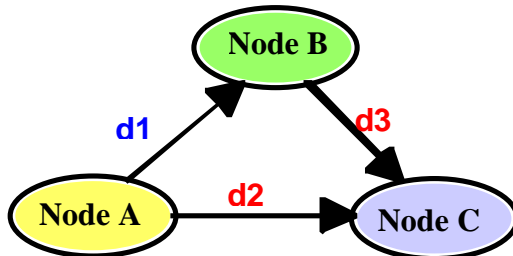


Figure 8 Three Node Timed Influence Net

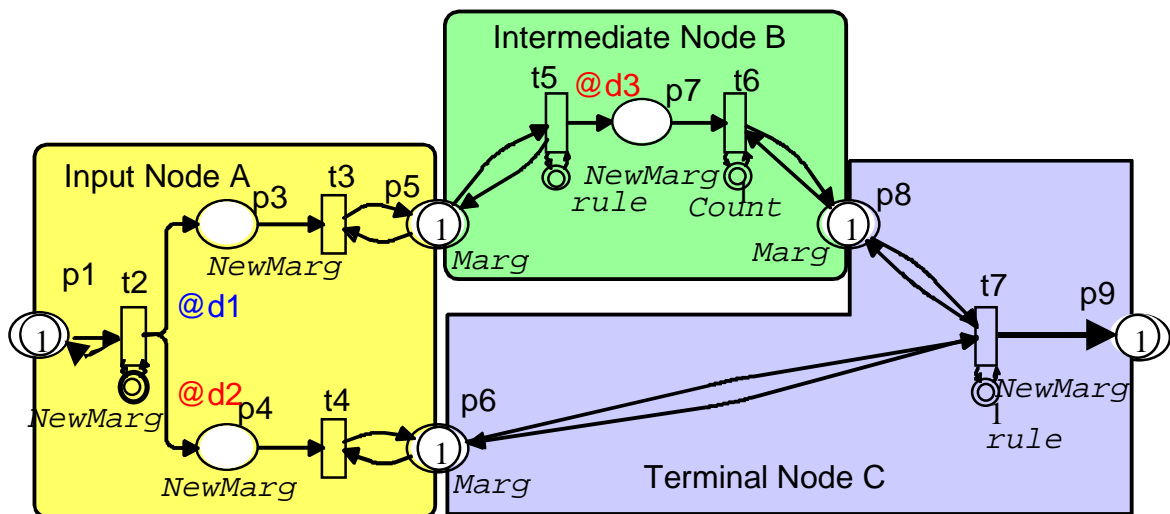


Figure 9 CP net of Influence Net

4 STATE SPACE ANALYSIS FOR VERIFICATION OF BEHAVIOR

State space analysis was used to verify the behavior of the CP net construction presented in Section 3. This section highlights key parts of the analysis and illustrates how the Occurrence Graph Analyzer (OGA) of Design/CPN can be used for behavior verification. The analysis was done in two steps. First a simple influence net model that has all three types of nodes was used and the corresponding CP net was created using Design/CPN. The OGA was used to generate occurrence graphs to generate all the possible occurrence sequences in the CP net. In addition, the state space report that provides statistical and standard properties of the CP net was used to verify its behavior. This analysis revealed that joins are a major factor in the behavior of the model. Thus, in the second step a detailed analysis of a join was conducted.

Figure 10 shows the timed influence net that was used for the verification process. Included are the conditional probability distributions for the intermediate and terminal nodes. In addition, the initial values of the conditional probabilities for each node is shown with the o subscript. The topology was selected because it has the minimal set of features needed to verify the behavior of larger nets. These features include multiple inputs and terminal nodes, an intermediate node, and at least one branch and one join.

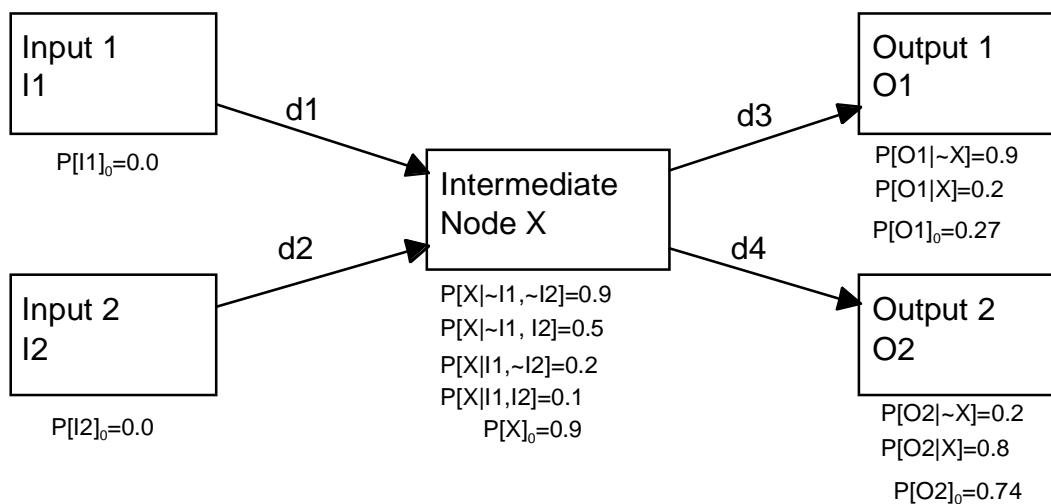


Figure 10 Verification Test Timed Influence Net

Occurrence graphs and state space reports were generated for four different cases. Each case illustrates, important features and characteristics of the discrete event model of the timed influence net. The cases are:

- 1.) Untimed (all delays set to zero and both input occur at time zero) single input,
- 2.) Timed unequal time delays and input time stamps of zero.
- 3.) Timed with all delays equal zero and unequal time stamps on the inputs.
- 4.) Timed with a combination of different time delays and time stamps on the inputs

To aid in understanding the occurrence graphs and the state space reports, the CP net of the timed influence net is shown in Figure 11. Most of the annotations have been suppressed; the names of the places and transitions are shown in Helvetica bold and the color sets are in italic.

The occurrence graph of the untimed single input (I1) case is shown in Figure 12. Details of the states and occurrences have been provided for only a few of the key nodes and arcs of the graph, and a level index has been added to help in the description of the behavior. From the topology of the CP net, we can see that with only input I1, there should be seven steps for the input to propagate its effect through the network, ending in a dead marking. The first three transitions to fire should be $t_{Input01}$, $t_{Input01}$, and t_{I1} because these transitions will be the only enabled transitions in this sequence. After t_{I1} fires, both t_{u1} and t_{um} are enabled. This results in a branch in the occurrence graph at node 4. The remaining steps of firing occur as t_{u1} , t_{um} , t_{dO1} , and t_{dO2} fire. The exact order of

firing is not specified in the CP net, hence there are several different paths from node 4 to the final node 12 in the occurrence graph. Observing the state of the CP net at node 12 reveals that the markings of the ResultO1 and ResultO2 places are correct given the input. ResultO1 shows the initial probability of 0.27 and a final probability of 0.76 which are the correct values. Similarly ResultO2 has probabilities of 0.74 and 0.32, which are also the correct values.

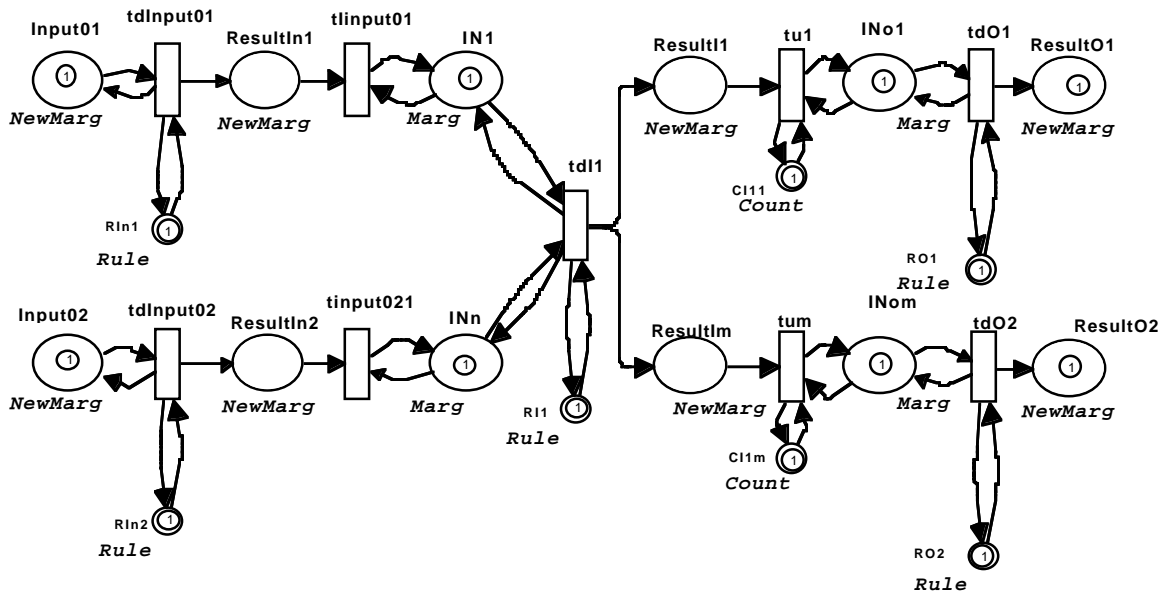


Figure 11 CP Net of Timed Influence Net

Design/CPN is capable of generating a state space report without executing the net. This report can be generated very quickly, even for large state spaces, and provides a great deal of useful information about the properties of the CP net that characterize the behavior of the net for a given marking. The report is divided into four sections: Statistical Information, Boundedness Properties, Home and Liveness Properties, and Fairness Properties. The Statistical section of the State Space Report for the Single Input case is shown in Table 1.

Table 1 Statistical Information For Single Input

Statistics	
Occurrence Graph	Scg Graph
Nodes: 12	Nodes: 12
Arcs: 15	Arcs: 15
Secs: 0	Secs: 0
Status: Full	

The statistics show that the state space has 12 states with 15 arcs. The Secs indicates the number of seconds it took for the processor to generate the State Space Report. In this case it was less than one second. The status means that the full state space has been analyzed. The SCC Graph stands for Strongly Connected Component Graph. The SCC graph has the same number of nodes as the state space. Since none of the SCCs has more than one node, there are no cycles in the state space, and, therefore, no infinite occurrence sequences. This is the correct characteristic for the CP net.

The Boundedness Properties of the report are shown in Table 2. This report lists the upper and lower integer bounds for each place in the net. These bounds indicate the maximum and minimum number of tokens that can exist in each place in the CP net in all the reachable markings. This report indicates that the CP net is functioning properly. Each of the *Counter* and *Rule* places always has one token as do the places that hold the current marginal probability values of the nodes of the influence net. The *Result* places hold either zero, one or two tokens depending on their location in the net. The

list of all the possible probability values of these place, and thus the maximum and minimum value of any possible probability profiles for the given initial marking.

Table 2 Upper Integer Bounds for Single Input Case

Boundedness Properties					
Best Integers Bounds	Upper	Lower	Best Integers Bounds	Upper	Lower
CI11 1	1	1	RIn2 1	1	1
CI1m 1	1	1	RO1 1	1	1
IN1 1	1	1	RO2 1	1	1
INn 1	1	1	ResultI1 1	1	0
INo1 1	1	1	ResultIm 1	1	0
INom 1	1	1	ResultIn1 1	1	0
Input01 1	1	1	ResultIn2 1	0	0
Input02 1	1	1	ResultO1 1	2	1
RI1 1	1	1	ResultO2 1	2	1

Table 3 Upper Multi-set Bounds for Single Input Case

Best Upper Multi-set Bounds

CI11 1	$1^1 + 1^2$
CI1m 1	$1^1 + 1^2$
IN1 1	$1^1(0.0,0) + 1^1(1.0,0) + 1^1(1.0,1)$
INn 1	$1^1(0.0,0)$
INo1 1	$1^1(0.2,0) + 1^1(0.2,1) + 1^1(0.9,0)$
INom 1	$1^1(0.2,0) + 1^1(0.2,1) + 1^1(0.9,0)$
Input01 1	$1^1(1.0,0) + 1^1(1.0,1)$
Input02 1	$1^1(1.0,0)$
RI1 1	$1^1(1,[0.9, 0.5, 0.2, 0.1] , [0, 0]) + 1^1(2,[0.9, 0.5, 0.2, 0.1] , [0, 0])$
RIn1 1	$1^1(1,[],[0]) + 1^1(2,[],[0])$
RIn2 1	$1^1(1,[],[0])$
RO1 1	$1^1(1,[0.9, 0.2] , []) + 1^1(2,[0.9, 0.2] , [])$
RO2 1	$1^1(1,[0.2, 0.8] , []) + 1^1(2,[0.2, 0.8] , [])$
ResultI1 1	$1^1(0.2,1)$
ResultIm 1	$1^1(0.2,1)$
ResultIn1 1	$1^1(1.0,1)$
ResultIn2 1	empty
ResultO1 1	$1^1(0.27,0) + 1^1(0.76,1)$
ResultO2 1	$1^1(0.32,1) + 1^1(0.74,0)$

Table 4 shows the lower multi-set bounds for the net. A lower multi-set bound is the largest multi-set which is smaller than all of the reachable markings of a place. In other words, it is a multi-set of tokens in a place whose values never change in all of the reachable markings. There are five such places in the Single Input Case. The first three in the table are associated with the Input I2 that was never activated so the markings of the place named INn1 and the Rule place named Rin2 never changed. The other Rule and counter places change because the counter values change. The initial marking of the ResultO1 and ResultO2 places never change because there are no output arcs from either of these places.

The third part of the state space report provides information about the Liveness Properties of the CP net as shown in Table 5. It lists dead markings and dead transitions. A dead marking is a marking with no enabled transition, a final state in the occurrence graph. The report shows that there is a single dead marking of node 12 in the occurrence graph. A dead transition is a transition that is never enabled from any reachable marking. In this CP net there are two dead transitions, tInput02 and tinput021. These transitions are on the path from the second input I2 and can never fire given the initial marking.

Table 4 Best Lower Multi-set Bounds

INn 1	1^(0.0,0)
InputO2 1	1^(1.0,0)
RIn2 1	1^(1,[],[0])
ResultO1 1	1^(0.27,0)
ResultO2 1	1^(0.74,0)
All others	empty

Table 5 Liveness Properties

Dead Markings: [12]
Dead Transitions Instances:
tdInputO2 1
tinputO21 1

The state space analysis shows that the CP net is operating properly with a single input. The next step is to generalize the analysis for multiple inputs with the net both untimed and timed.

The most general case is that with all inputs present and the network untimed. This case can be created by marking all inputs with tokens that enable the input transitions and setting all of the delay values to zero. This initial marking will provide the maximum flexibility in firing sequences for the transitions in the CP net. It will also provide the largest state space for the CP net. Any initial marking that incorporates at least one delay that differs from all the other delays, or incorporates a different time stamp on any input token will provide a restriction on the firing sequences and generate a subset of the state space of the "untimed" CP net.

The state space report revealed that the untimed CP net with both inputs activated has the same behavioral properties as the single input case. The statistical report showed that the SCC graph and the occurrence graph have the same number of nodes (122) and arcs (235), and, therefore, no cycles. The integer boundedness properties are also the same as for the single input case, with the exception that ResultI1 and ResultIm can have two tokens as the upper integer bound and ResultO1 and ResultO2 can have three tokens as the upper bound.

The topology of the 122 node occurrence graph revealed several important characteristics of the untimed CP net. First, there were three final states in the graph and 15 levels in the graph. Two of the final states (nodes 121 and 122) are reached if all 15 levels are traversed while the third final state (node 87) requires only 10 levels. At the first level, it is possible to reach all three final states. However choices made at level 1 or level 2 could eliminate one final state as a reachable marking. Level 3 is the last level where it is possible to reach more than one final state. It is the branches in the influence net that cause these choices.

Most of the boundedness properties were the same as for the single input case. The key property is associated with the ResultO1 and ResultO2 places. The portion of multi-set boundedness properties associated with the two terminal nodes is shown in Table 6. The probability and the counter values of the tokens shows that each place starts with its initial value, (0.27, 0) and (0.74, 0), respectively. These are the initial values because the counter values are zero. The remaining elements of the multi-set show that ResultO1 can have values of (0.55, 1), (0.76, 1), (0.83, 1) and (0.83, 2). The token with the highest counter value is the final state of the node, 0.83, which is the correct value, given both inputs. The other values are intermediate values that are computed when either the first input propagates through the net before input 2 or when the second input propagates through the net before input 1. More precisely, these values correspond the sequence of arrival of the input updates at the transition tdI1 (see Figure 11) which computes the new marginal probability of node X. The value (0.83, 1) indicates that the input updates were both available to the transition tdI1 when it fired for the first time. Notice that if the updates do not arrive simultaneously then transition tdI1 will fire twice, once for each update.

Table 6 Boundedness Properties of the Untimed CP Net

Best Upper Multi-set Bounds

ResultO1 1	1^(0.27,0) + 1^(0.55,1) + 1^(0.76, 1) + 1^(0.83, 1) + 1^(0.83, 2)
ResultO2 1	1^(0.26,1) + 1^(0.26,2) + 1^(0.32, 1) + 1^(0.5,1) + 1^(0.74, 0)

Best Lower Multi-set Bounds

ResultO1 1	1^(0.27,0)
ResultO2 1	1^(0.74,0)
All Others	empty

The Liveness properties indicate that there are three dead markings, nodes 87, 121, and 122 of the occurrence graph. Unlike the single input case, there are no dead transitions because each transition can at fire least once on some path through the occurrence graph. In our case, each transition fires at least once on all paths through the occurrence graph.

The value of the ResultO1 and O2 places for dead markings is shown in Figure 13. The other places for the dead markings of nodes 121 and 122 are the same as for node 87 with the exception that the counter values for nodes 121 and 122 are three in all places with counters.

The three dead markings are consistent with the State Space Report and desired behavior of the CP net. Node 87 is the result of firing sequences in which the two updates for Input 1 and Input 2 occur before the transition tdII fires, hence it only fires once with the correct marginal probability of node X. Thus, the markings of the places ResultO1 and ResultO2 make one single change from the initial probability to the final probability. Occurrence Graph node 121 is reached if transition tdII fires when enabled by the update from input 1 without the update from input 2 and then fires a second time when the update from input 2 arrives, thus generating two tokens. The first token generates an intermediate marginal probability values of 0.76 and 0.32 in ResultsO1 and O2, respectively, before the second token generates the final marginal probability values. Occurrence graph node 122 has the same property except the result of input 2 occurs first before the final result that includes both inputs.

87			
ResultO1 1:	$1^{(0.27,0)}$	$@[0]^+$	$1^{(0.83,1)}$
ResultO2 1:	$1^{(0.26,1)}$	$@[0]^+$	$1^{(0.74,0)}$
121			
ResultO1 1:	$1^{(0.27,0)}$	$@[0]^+$	$1^{(0.76,1)}$
			$@[0]^+$
			$1^{(0.83,2)}$
ResultO2 1:	$1^{(0.26,2)}$	$@[0]^+$	$1^{(0.32,1)}$
			$@[0]^+$
			$1^{(0.74,0)}$
122			
ResultO1 1:	$1^{(0.27,0)}$	$@[0]^+$	$1^{(0.55,1)}$
			$@[0]^+$
			$1^{(0.83,2)}$
ResultO2 1:	$1^{(0.26,2)}$	$@[0]^+$	$1^{(0.5,1)}$
			$@[0]^+$
			$1^{(0.74,0)}$

Figure 13 ResultO2 and O2 for the Three Dead Markings

This explanation demonstrates the need for the counters that enforce the FIFO protocol. Without this protocol it would not be possible to distinguish the final value of the marginal probability from the intermediate values.

Once the behavior of the untimed CP net was verified, the behavior of the timed net was examined. There are two types of timing constraints that can be incorporated in the CP net. The first is by setting the timing of the inputs, and the second is by incorporating time delays on the arcs between parents and children. Each was investigated individually and then in combination.

The occurrence graphs clearly showed the effects of timing. The occurrence graph with the Input 1 and 2 occurring at $t = 0$ and $t = 1$, respectively, showed that separating the inputs reduces the size of the state space from 122 nodes to 23 and reduces the dead markings from three to one. As expected, the occurrence graph is simply two single input occurrence graphs that are concatenated.

The occurrence graphs with obtained different time delays and both inputs with time stamp of zero, showed that these delays have the impact of separating the inputs but only after both input transitions have fired. This is because either input can be concurrently enabled throughout every step of the propagation the other while the simulation time remains at zero. Thus, the first half of the occurrence graph has many more nodes than that of the case where the inputs were timed. Nevertheless the final results with respect to the dead marking are the same.

Finally, a mix of time delays and time difference of the inputs was examined. These can cause the occurrence graph become a single chain as all choice is eliminated.

As was noted earlier when the occurrence graph of the untimed net was introduced, the determination of which dead marking will be reached occurs early in any firing sequence. A closer look at the occurrences in the occurrence graph reveals that it is the joins in the influence net that determines the number of dead markings in an untimed CP net model the influence net. A net with

no joins will have a single dead marking. A net with at least one join will have multiple dead markings. Branches have no impact on the number of dead markings, but do impact the size of the state space.

To further examine the impact of joins on the number and composition of dead markings, a simple CP net model of a join was created as shown in Figure 14. This CP net represents a three legged join. To keep the model simple, only the places and transitions that update each leg of the join were modeled. The join has a single output because branches do not effect dead markings.

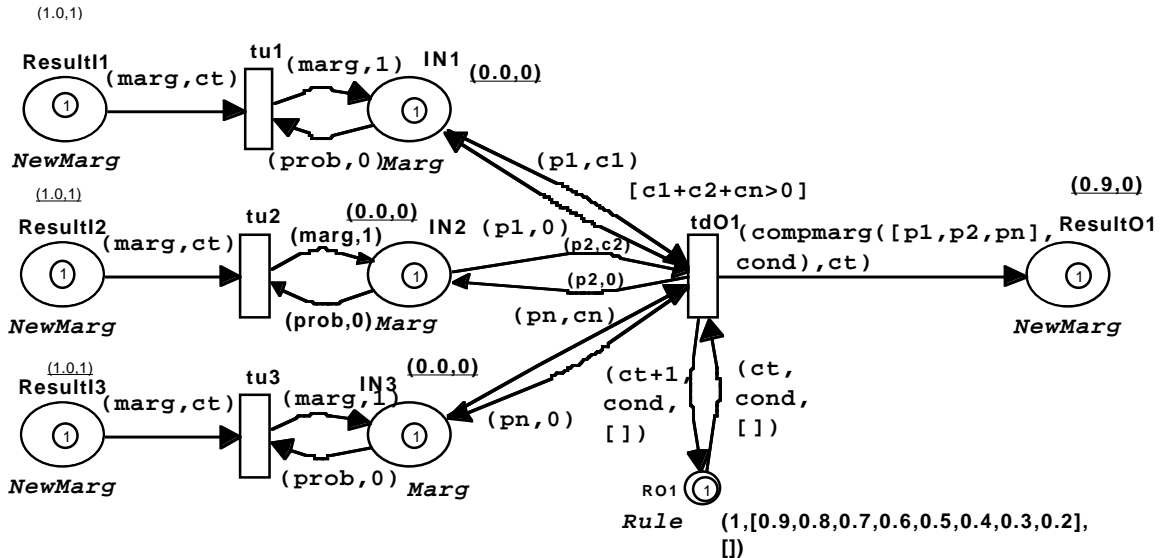


Figure 14 CP Net of Three Legged Join

The behavior of this CP net model can be anticipated. Initially transitions tu1, tu2, and tu3 will be concurrently enabled. Because there is no control on the firing sequences of concurrently enabled transitions, any one of those three can fire first. As soon as one fires, transition tdO1 becomes concurrently enabled with the remaining two tui transitions. It can fire next or wait until one or more of the two remaining tui fire. Whenever tdO1 fires, it "reads" the current probability values of each preset place (IN1, IN2, and IN3), and generates a token that is a double with the "new" marginal probability based on those values plus the current counter number. The values of the preset places depend on whether the corresponding tui has fired. Depending on the firing sequence, transition tdO1 can fire only once if it "waits" until all three tui have fired, it can fire twice, if it fires once before all three tui have fired or it can fire three times if it fires sequentially after each tui fires.

Obviously there are a number of potential firing sequences for this CP net model. We can view the combination of potential firing sequences based on all of the possible sequences of arrivals of the three inputs. For three inputs there are 13 such sequences. Each sequence will produce a different marking in the Result place, hence we can expect the occurrence graph of the CP net to have 13 dead markings.

The values probability values generated in the firing sequences can be predicted.

Let $P[X]$ be the marginal probability of the join node that is calculated when tdO1 fires.

Let u_i be the current (old) marginal probability value of the i^{th} parent.

Let u_i' be the new, updated marginal probability value of the i^{th} parent

Then, for the 13 potential sequences of the 3 inputs there are $2^3 = 8$ marginal probability values:

$$P[X|u_1, u_2, u_3], P[X|u_1, u_2, u_3'], P[X|u_1, u_2', u_3], P[X|u_1', u_2, u_3], P[X|u_1, u_2', u_3'], P[X|u_1', u_2, u_3'], P[X|u_1', u_2', u_3], P[X|u_1', u_2', u_3']$$

These sets of values can be arranged in a partial order based on the number of new updates in the condition.

Without loss of generality, the initial values of the old and new marginal probabilities and the conditional probability distribution have been chosen in a way that will simplify the analysis of the state space. The initial marginal probability values are set to zero and the new updates are all one.

The conditional probability distribution is a set of eight unique values starting with 0.9 and descending in equal increments to 0.2. Because the calculation of marginal probability involves products of the input probabilities and their compliments, each input sequence will uniquely result in one of the values of conditional probability distribution.

These characteristics are summarized in Table 7. The first column is an index indicating the sequence number. The second three columns indicate the order of processing of the three updates, u_1 , u_2 , and u_3 . For example 1, 2, 3 means the updates are processed in sequence, while 2, 1, 1 means that u_2 and u_3 are processed first, simultaneously followed by u_1 . This ordering corresponds to different paths through the input space that was illustrated in Figure 1. The fifth column lists, in sequence, the marginal probability values that are computed for the specific sequence. The sixth column provides the actual probability values for the set of initial markings as discussed in the previous paragraph.

The behavior described in Table 7 can also be presented graphically as shown in Figure 15. The nodes in the graph are triples that indicate the value of the three inputs that are used in the marginal probability calculation. For example, (0, 1, 0) means $u_1=0$, $u_2=1$ and $u_3=0$. The arcs indicate an allowable change from one value to another. The initial conditions start with the input (0, 0, 0), and the input sequence always terminates with all values of the inputs equal to one (1, 1, 1). There are 13 paths from the initial node to the terminal node. Each path presents a feasible sequence of changes in the marginal probability. Note that there are arcs from the initial node to all other nodes and arcs from all nodes to the terminal node.

Table 7 Sequences of Updates

No.	u_1	u_2	u_3	Marginal Probability Calculations	Values
1	1	2	3	$P[X u_1', u_2, u_3]$, $P[X u_1', u_2', u_3]$, $P[X u_1', u_2', u_3']$	0.5, 0.3, 0.2
2	1	3	2	$P[X u_1', u_2, u_3]$, $P[X u_1', u_2, u_3']$, $P[X u_1', u_2', u_3']$	0.5, 0.4, 0.2
3	2	1	3	$P[X u_1, u_2', u_3]$, $P[X u_1', u_2', u_3]$, $P[X u_1', u_2', u_3']$	0.7, 0.3, 0.2
4	2	3	1	$P[X u_1, u_2, u_3']$, $P[X u_1', u_2, u_3']$, $P[X u_1', u_2', u_3']$	0.8, 0.4, 0.2
5	3	1	2	$P[X u_1, u_2', u_3]$, $P[X u_1, u_2', u_3']$, $P[X u_1', u_2', u_3']$	0.7, 0.6, 0.2
6	3	2	1	$P[X u_1, u_2, u_3']$, $P[X u_1, u_2', u_3']$, $P[X u_1', u_2', u_3']$	0.8, 0.6, 0.2
7	1	2	2	$P[X u_1', u_2, u_3]$, $P[X u_1', u_2', u_3']$	0.5, 0.2
8	2	1	1	$P[X u_1, u_2', u_3']$, $P[X u_1', u_2', u_3']$	0.6, 0.2
9	1	1	2	$P[X u_1', u_2', u_3]$, $P[X u_1', u_2', u_3']$	0.3, 0.2
10	2	2	1	$P[X u_1, u_2, u_3']$, $P[X u_1', u_2', u_3']$	0.8, 0.2
11	1	2	1	$P[X u_1', u_2, u_3']$, $P[X u_1', u_2', u_3']$	0.4, 0.2
12	2	1	2	$P[X u_1, u_2', u_3]$, $P[X u_1', u_2', u_3']$	0.7, 0.2
13	1	1	1	$P[X u_1', u_2', u_3']$	0.2

The occurrence graph of the three legged join CP Net had 51 nodes, 58 arcs, and 13 dead markings. The other properties are consistent with the state space analysis performed on the untimed CP net of Figure 11. The most interesting property is the upper multi-set bound for the ResultO1 node. This multi-set is $1^{\setminus}(0.2,1) + 1^{\setminus}(0.2,2) + 1^{\setminus}(0.2,3) + 1^{\setminus}(0.3,1) + 1^{\setminus}(0.3,2) + 1^{\setminus}(0.4,1) + 1^{\setminus}(0.4,2) + 1^{\setminus}(0.5,1) + 1^{\setminus}(0.6,1) + 1^{\setminus}(0.6,2) + 1^{\setminus}(0.7,1) + 1^{\setminus}(0.8,1) + 1^{\setminus}(0.9,0)$.

This upper multi-set bound shows eight potential marginal probability values for the three legged join. These are the same values shown in Table 7. The 13 markings for ResultO1 extracted from the dead markings are shown in Table 8. Each row of the table represents one of the 13 paths from the initial to the final value and each path contains the sequence of probability values that can occur in a probability profile.

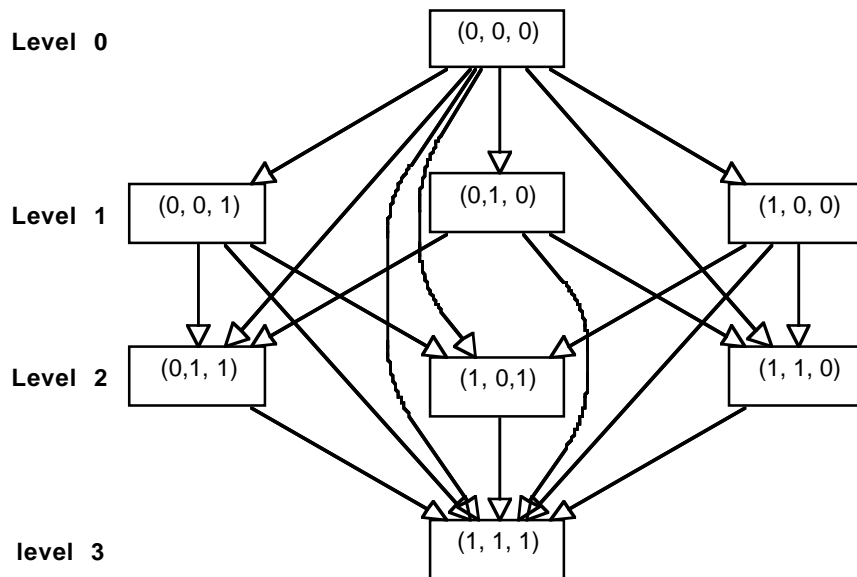


Figure 15 Graph of 13 Sequences Given 3 Inputs

Table 8 Dead Markings for the ResultO1 Place

OG Node	Counter = 0	Counter = 1	Counter = 2	Counter = 3
25	0.9	0.2		
37	0.9	0.3	0.2	
38	0.9	0.4	0.2	
35	0.9	0.5	0.2	
42	0.9	0.6	0.2	
40	0.9	0.7	0.2	
44	0.9	0.8	0.2	
46	0.9	0.5	0.3	0.2
47	0.9	0.5	0.4	0.2
48	0.9	0.7	0.3	0.2
49	0.9	0.7	0.6	0.2
50	0.9	0.8	0.4	0.2
51	0.9	0.8	0.6	0.2

If one or more of the inputs to the three legged join are given a time stamp that is different from the other inputs, the impact should be to reduce both the size of the state space and the number of dead markings. This is because the time difference in the inputs separates the availability of the tokens and reduces the number of transitions that can be concurrently enabled.

This behavior was clearly depicted in the occurrence graph of the three legged join with inputs 1 and 2 with time stamps of zero and input 3 with a time stamp of 2. In each dead marking, the initial marginal probability value of the node was 0.9 with a counter value of zero. All three dead markings showed the correct final probability value of 0.2 that occurs at time equal 2. The three dead markings differed in the set of markings at time equal zero. Each has the same final probability value at $t = 0$ but each shows a different intermediate value caused by a different path through a portion of the output space. The intermediate probability values occur because of different firing sequences due to concurrent enablements of transitions. These intermediate values with the same time stamp are artifacts of the updating process and have no real meaning. The FIFO protocol ensures that tokens produced at joins will be processed by children of the join in the order they were created, with the last token value being the correct and final value of the update process at a given point in time. The sequenced set of correct and final probability values of a node constitute a probability profile for the node for a given input. In the example, each of the dead markings generates an identical probability

profile starting with the initial value of 0.9 that changes to 0.3 at time $t = 0$ and remains at that value until $t = 2$ when it changes to 0.2.

Figure 16 shows the output space for the join. The arcs are labeled with the updated probability that arrived at the join that triggered the transition in state. The bolded nodes and arcs indicate the portion of the state space that is reachable for this timed example.

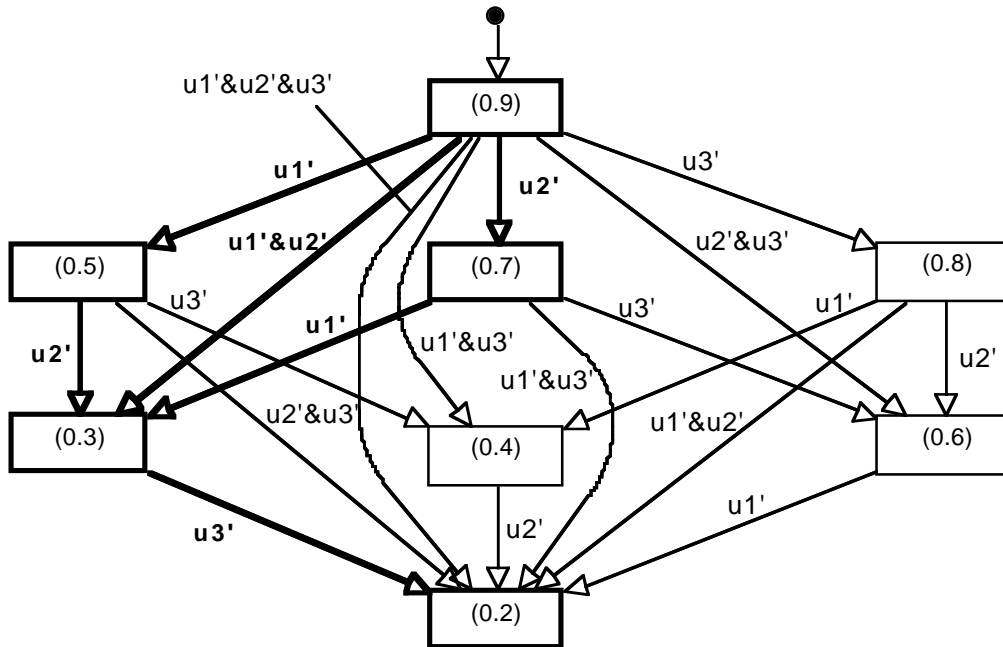


Figure 16 Output Space of ResultO1

This example can be used to illustrate the concept of the probability profile. If the arrival times of the inputs to the join can be selected, it is possible to control the probability profile. Figure 17 shows two profiles that can be generated from the join. The profile with the square points is for the case where input 2 arrives at time = 1, input 3 arrives at time = 2 and input 1 arrives at time = 4. If the arrival times of the inputs are changed to inputs 1, 2 and 3 at times 1, 1, and 2, respectively, the probability profile changes to the one denoted by the circles. If the objective was to cause the output probability to be as small as possible, the second profile is preferred over the first because it decreases more rapidly. While it is easy to determine the probability profile of a simple net, the state space analysis has shown that the CP net can be used to generate the probability profile for any influence net with time delays and a specific set of timed inputs.

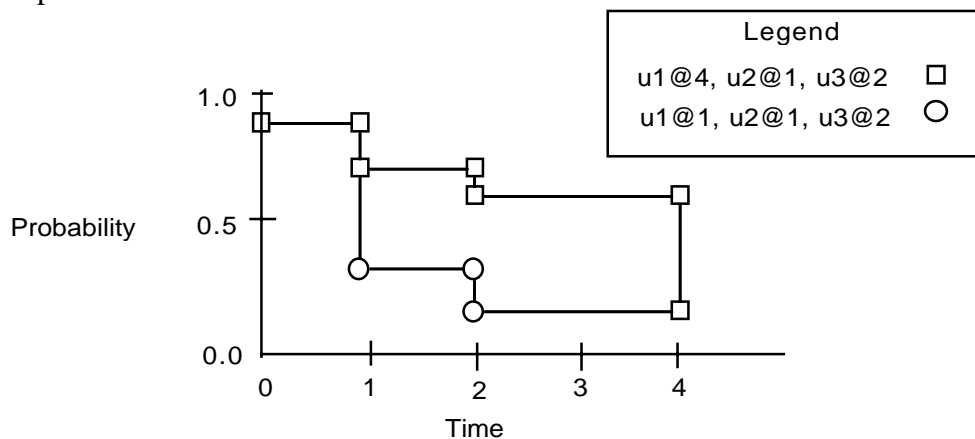


Figure 17 Example of a Probability Profile for the Join

5 SUMMARY AND FUTURE DIRECTIONS

We have introduced the formal concepts that permit the introduction of time to influence nets. Influence nets are basic static equilibrium models. They have no concept of time. It was shown that in order to correctly capture the behavior of an influence net with time delays, they must be converted to a discrete event system model. The most general model of discrete event systems, the colored Petri net, is a suitable modeling formalism. Three similar CP nets that can model the behavior of input, output, and intermediate nodes of the net have been proposed. A method of interconnecting the three types of CP nets to form a complete model of the timed influence net was demonstrated. State space analysis of the CP net components was conducted to analyze their behavior. This analysis demonstrated that the CP net construct behaves correctly and can be used with confidence to model any timed influence net.

Future research areas include developing ways to use the CP net models to support effects based planning and dynamic re-planning of actions that support Courses of Action. We plan to investigate methods for using these models to track and measure progress of actions as situations unfold. The goal is to determine how the information produced by these models on the changing potential for achieving desired effects and results can be used in dynamic re-planning.

ACKNOWLEDGEMENTS

The authors wish to thank the anonymous referees for their review and comments which significantly improved the quality and accuracy of this paper.

REFERENCES

- [1] Cassandras, C. G. (1993), *Discrete Event Systems, Modeling and Performance Analysis*, Aksen Associates Incorporated, Boston MA.
- [2] Friedman, N and Goldszmidt, M. (1996), Learning Bayesian Networks with Local Structure, *Uncertainty in Artificial Intelligence: Proceedings of the 12th Conference*, Morgan Kaufmann.
- [3] Jensen, F. V. (1996) *An Introduction of Bayesian Networks*, UCL Press limited, London
- [4] Jensen K. (1992). *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use*. Springer-Verlag, Berlin, Germany.
- [5] Jensen K. (1997). *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use. Volumn 1. Basic Concepts*. Monographs in Theoretical Computer Science, Springer-Verlag, Berlin, Germany.
- [6] Jensen K. (1997). *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use. Volumn 2. Analysis Methods* Monographs in Theoretical Computer Science, Springer-Verlag, Berlin, Germany.
- [7] Jensen K. (1997). *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use. Volumn 3. Practical Use*. Monographs in Theoretical Computer Science, Springer-Verlag, Berlin, Germany.
- [8] Kristensen, L. M., Christensen, S, and Jensen K. (1998) *The Practitioner's Guide to Coloured Petri Nets*, Software Tools for Technology Transfer.
- [9] Murata, T. (1989). Petri Nets: Properties, Analysis and Applications, *Proceedings of the IEEE*, Vol. 77, No. 4, April 1989.

- [10] Pearl, J. and Verma, T. S. (1991), A Theory of Inferred Causation, in Allen, J., Fikes, R. and Sandewall, E. (Eds.) *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference*, San Mateo, CA Morgan Kaufmann.
- [11] Rosen, J. A. and Smith, W. L. (1996) *Influence Net Modeling with Causal Strengths: An Evolutionary Approach*, Proceedings of the Command and Control Research and Technology Symposium, Naval Post Graduate School, Monterey CA, June 25-28, 1996.
- [12] Wagenhals, L. W., Shin, I, and Levis, A. H. (1998) Creating Executable Models of Influence Nets with Colored Petri Nets, *International Journal for Software Tools for Technology Transfer*, vol. 2 no. 2, pp 168-181, Springer-Verlag.