Extension of Petri Nets for Representing and Reasoning with Tasks with Imprecise Durations

Stanislav Kurkovsky[1] and Rasiah Loganantharaj[2]

Department of Computer Science[1], Columbus State University, Email: kurkovsky_stan@colstate.edu
Intelligent System Laboratory[2], Center for Advanced Computer Studies
University of Louisiana at Lafayette, Email: logan@cacs.louisiana.edu

Abstract

This paper presents an extension of Petri net framework with imprecise temporal properties. We use possibility theory to represent imprecise time by time-stamping tokens and assigning durations to firing of the transitions. A method for approximation of an arbitrary temporal distribution with a set of possibilistic intervals is used to introduce the composition operation for two possibilistic temporal distributions. We developed a method to determining an effective enabling time of a transition with incoming tokens with possibilistic distributions. The utility of the proposed theory is illustrated using an example of an automated manufacturing system. The proposed approach is novel and has a broad utility beyond a timed Petri network and its applications.

Keywords

Petri nets, possibility theory, possibilistic temporal intervals, temporal composition.

1. Introduction

A careful examination of many real world applications reveals that they involve synchronization of several tasks and the durations of activities must be appropriately modeled, as opposed to representing them by fixed quantities. Suppose, a machine C requires two units A and a unit of B to produce an assembled product ABA. The operation of machine C is said to be synchronized with the availability of two units of A and one unit of B. In real world applications, the duration of a task may vary and cannot be realistically modeled by a fixed duration.

A Petri network is an ideal representation for modeling synchronization among tasks so as to study the system's dynamic behavior and to prove some interesting properties such as deadlock and starvation. The framework of Petri nets does not explicitly provide the notion of time. Ordinary Petri nets may be viewed as operating according to some internal (system) clock. In this view, each firing of a transition is associated with one clock cycle or tick. The length of the interval between ticks is not important, only the relations "before" and "after" among ticks are important. The occurrence of an event (firing of a transition) is not dependent on the number of internal ticks since the start of the clock. It depends only on the configuration of the net and its marking at the given clock cycle. While a Petri network is useful, its inability to address duration of tasks and processes hinders its utility for broad classes of applications such as planning and scheduling. There have been numerous attempts to associate time with tokens, places, arcs and transitions and many of these approaches seem to be ad-hoc and have the limitation of using fixed time duration for tasks.

In this paper we will examine the previous work on timed Petri networks and systematically develop a method and the underlying theory to model tasks with imprecise duration and show how they can be used to model complex processes and how to reason with them. The approach we propose is general enough to be applicable for other modeling involving imprecise time duration and dependency. When an interval is presented with possibilistic distribution, it is important to develop a temporal composition operation for propagating tokens across transitions in an extended Petri network. We provide details of possibilistic temporal composition operation and show that it generalizes temporal composition of metric bounds.

The paper is organized as follows. We provide the background information of Petri networks in Section 2 and it is followed by a review of timed Petri networks presented in Section 3. In Section 4, we describe possibilistic representation of time, which is followed by Sections 5 and 6 discussing possibilistic Petri networks and algorithms for their execution. In Section 6, we provide a summary and a discussion.

2. Background Information on Petri Net Theory

Ever since the introduction of the Petri net theory by Petri in his Ph.D. dissertation, it has been widely used for modeling the dynamics of systems of the most diverse domains. A Petri net is a graph-based structure consisting of places and transitions. The net simulates the dynamic behavior of a system by continuously firing enabled transitions when tokens are removed and inserted into places.

Let us illustrate the application of Petri Nets using the following example of an automated manufacturing system shown in Figure 1. The system is composed of four machines (Machine 1, Machine 2, Machine 3, and Machine 4) and two robots (Robot 1 and Robot 2). The system can process two types of parts, namely A and B. Part A needs sequential processing on Machine 1 and then on Machine 2. Part B needs processing on Machine 3. Machine 4 takes two parts A and one part B and assembles them into the final product. Robot 1 and Robot 2 handle loading/unloading operations between Machine 1 and Machine 2. The graph of the corresponding Petri net is shown on Figure 2. The availability of part A and part B is modeled by tokens in places p_1 and p_5 respectively. The processing of Machine 1 is modeled by transition t_1, and its product after processing is modeled as a token in place p_2. When one token is available in place p_2, transitions t_2 and t_3 are enabled, but only one of them can fire. There is non-determinism in selecting which of these transitions should fire, which is usually resolved by randomly breaking a tie [25].

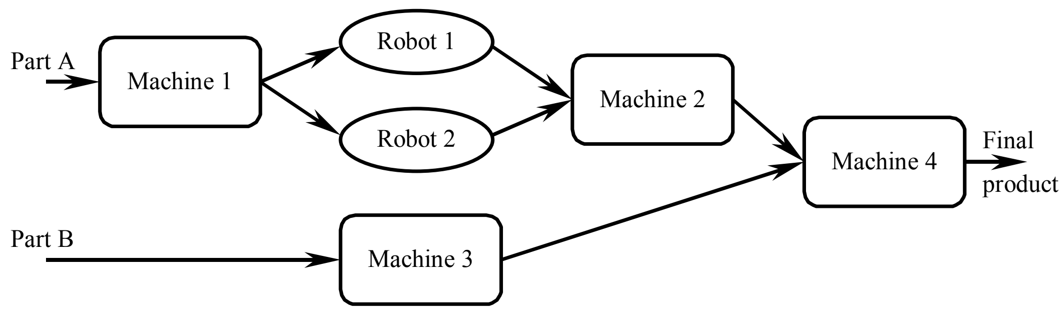

Figure 1. An automated manufacturing system

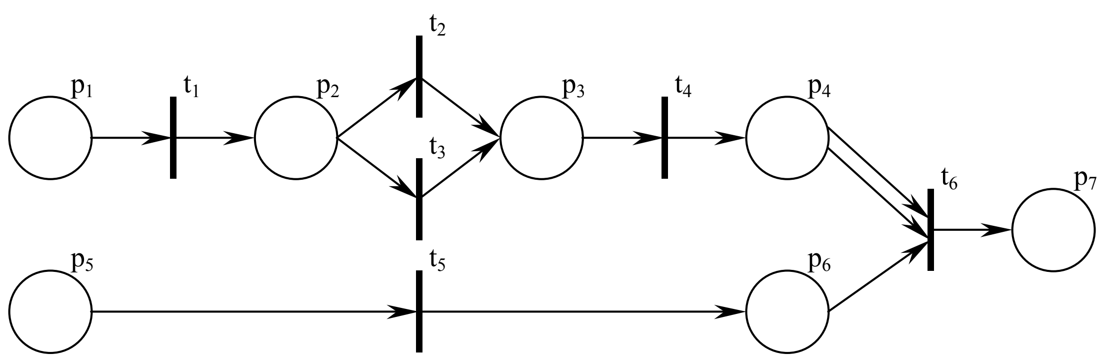

Figure 2. Petri net graph for the automated manufacturing system

In this representation, no information is given about the duration of operation of robots and machines. Alternatively, the operations are assumed instantaneous, which is not realistic. Also, the capacity of the buffers (places p_1 through p_7) is assumed to be infinite.

3. Review of Timed and Fuzzy Petri Networks

Qualitative notion of time is implicitly represented in Petri nets in the sense that each firing of a transition is associated with one clock cycle or tick. The interval length between ticks is not relevant, only the relations "before" and "after" among ticks need to be considered. The firing of a transition (which represents the occurrence of an event) is not dependent on the number of internal ticks since the start of the clock. It depends only on the configuration of the net and its marking at the given clock cycle.

To capture the quantitative notion of time, “external time” (or “external clock”) was introduced in [2]. In such a representation, the occurrence of the events depends not only on the marking, but also on the elapsed time since the occurrence of some other events. Having a single time line subsumes both internal and external times. There are several proposals incorporating the notion of time to virtually every component of the Petri nets framework, namely tokens, transitions, places, and arcs [3, 17, 20, 23].

Because of the fact that any Petri net evolves by moving tokens across places and enabling of transitions is dependent on the availability of tokens, it seems quite natural to associate time with tokens [15, 28, 30]. In most cases, enabling of a transition depends on the timestamps of tokens. This timestamp may be interpreted as the age of a particular token (how much time elapsed since it has been produced), or as the age of a sequence of tokens that are generated after a series of firing transitions.

Time can be associated with transitions too [4, 10, 28]. The possible interpretations of time θ being associated with transition t_j include:

- t_j may (or must) fire only after θ time units pass after t_j becomes enabled (θ is assumed to be relative);
- t_j may fire only during time interval θ (θ is assumed to be absolute);
- the duration of firing of t_j is θ (θ is assumed to be relative). In this case t_j may (or must) start firing as soon as it becomes enabled.

The first two approaches of associating time with transitions seem to be unnatural, because it is difficult to imagine a description of a system where certain processes can occur only within a given time frame. It is more likely that the processes taking place in a modeled system would have some length in time, which will be represented as durations of firing transitions of a Petri net.

Waiting time can be associated with places [8, 29]. Once a token has been added to a place, it will not contribute to enabling of any transition before the waiting time associated with that place has elapsed. Also, a transition must fire as soon as it becomes enabled.

Time can also be associated with arcs [19]. In this case, it is interpreted as a period of time that must elapse until a token will arrive from a place to a transition or vice versa. This representation is equivalent to representing time as the duration of firing a transition. Associating time with arcs and/or places instead of transitions simply changes the way in which a Petri net is interpreted. In each case (time associated with transitions, places or arcs) the semantics of a Petri net are defined in a similar way.

A wide range of extensions to the basic Petri net framework has been done to incorporate imprecisely-known information about the movement of tokens through the network [6, 7, 21, 30], most of which are outlined in [5]. These extensions include: fuzzy token locations, fuzzy firing rules (when transitions fire with a degree of possibility), fuzzy firing windows (during which transitions may fire), etc. In our approach, imprecise information is given about temporal properties of a dynamic system. In Petri nets, these properties are modeled by the timestamps associated with each token (absolute time of its generation) and durations of firing of the transitions.

4. Possibilistic Representation of Time

Dubois and Prade [13, 14] use possibility theory to model and manage temporal knowledge that involves imprecisely known information. This approach uses points as the temporal primitive. Imprecisely known dates are modeled by a fuzzy set with a unimodal possibility distribution over the temporal axis. Fuzzy temporal intervals are derived from fuzzy dates that limit the time span during which an event occurs. Fuzzy durations are treated in a similar way. A typical possibilistic temporal interval representing a fuzzy duration may be approximated by a trapezoidal shape (Figure 3). A temporal interval shown in Figure 3, which is represented as $\{a'(0), a(1), b(1), b'(0)\}$, has a duration certainly between a and b and most plausibly between a' and b'. The approach proposed by Dubois and Prade allows for

- Representation of uncertain precedence relations (such as "much before" or "closely after") between events,
- Comparison, ranking and ordering of fuzzy dates and durations, and
- Propagation of qualitative fuzzy temporal constraints.

The approach proposed by Dubois and Prade establishes a very solid foundation for applying possibility theory for temporal reasoning by providing a possibilistic representation of temporal primitives.

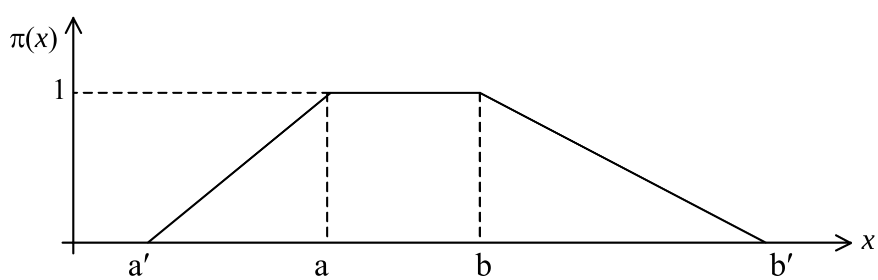

Figure 3. Trapezoidal approximation of a possibilistic interval

Trapezoidal approximation of possibilistic distributions does not provide enough flexibility for modeling of mathematical functions. They are overly restrictive to the resulting function and are hardly useful for piece-wise approximation. For the purposes of modeling functions by approximation and to minimize the number of elementary intervals participating in the approximation, we propose to represent possibilistic distributions using *alternative trapezoidal shapes* (Figure 4). It is different from the trapezoids presented earlier – alternative trapezoids have their vertical edges parallel to each other, lower edge coincides with the horizontal axis, and the last edge (slope) is arbitrary. Such an alternative trapezoidal shape T is determined by four parameters:

$$T = \{a, b, h_1, h_2\}.$$

Depending on the inclination of the slope (i.e. the sign of ($h_1 - h_2$)) we will distinguish between L-trapezoidal ($h_1 < h_2$) and R-trapezoidal ($h_1 > h_2$) shapes. Instances of these trapezoidal shapes include rectangular shape ($h_1 = h_2$), L-triangular ($h_1 = 0$) and R-triangular ($h_2 = 0$) shapes.

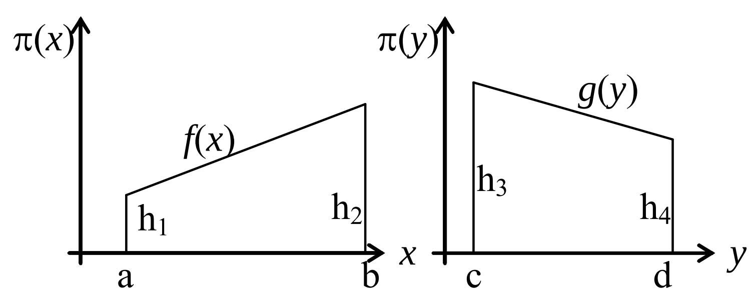

Figure 4. L-trapezoidal ($h_1 < h_2$) and R-trapezoidal ($h_3 < h_4$) possibilistic distributions

It is possible to approximate any function using only rectangular shapes, but a better approximation will require a very fine quantification and therefore will result in many elementary rectangles used to approximate the function. Approximations using several trapezoidal shapes are closer to the real functions and require less elementary shapes participating in such approximations (Figure 5).

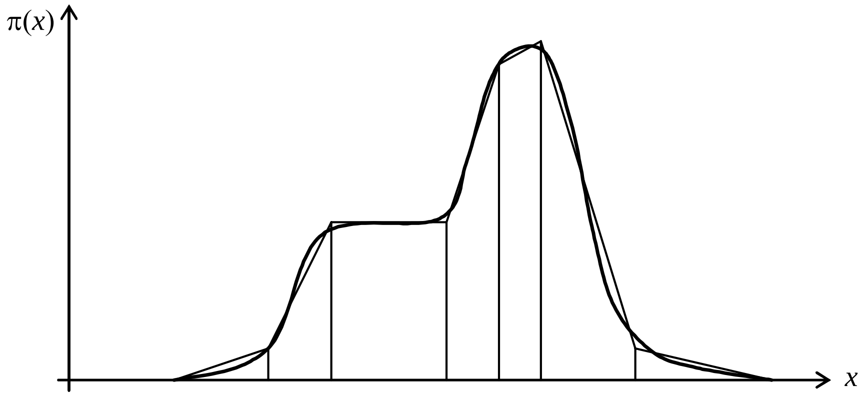

Figure 5. Approximation of a function using L- and R-trapezoidal shapes

An arbitrary function *f(x)* representing a possibilistic distribution may be modeled as a disjunction of *k* elementary distributions $m_i(x)$.

$$f(x) = \bigcup_{i=1}^{k} m_i(x)$$

In general, a possibilistic distribution can be represented as a sequence of *n+1* adjacent points and their corresponding possibility measures. It takes the form of

$$\pi(x) = [x_0(\pi(x_0)), x_1(\pi(x_1)), x_2(\pi(x_2)), \ldots, x_n(\pi(x_n))] = [\{x_i(\pi(x_i)), \forall\, i = 0,\ldots n\}],$$

where x_i is a point and $\pi(x_i)$ is a possibility measure of x_i. We assume that the value of possibility between two adjacent points x_i and x_{i+1} linearly varies from $\pi(x_i)$ to $\pi(x_{i+1})$ or remains constant (in this case $\pi(x_i)$ must be equal to $\pi(x_{i+1})$). If the function $f(x)$ is modeled by a disjunctive possibility distribution $\pi(x)$, the corresponding elementary possibilistic distributions can be constructed as

$$m_i(x) = [x_i(\pi(x_i)), x_{i+1}(\pi(x_{i+1}))], \forall\, i = 0, 1, 2, \ldots, n-1.$$

It is clear that such representation will result in *n* elementary trapezoidal shapes, union of which models the entire possibilistic distribution $\pi(x)$.

Let us introduce a formal representation of the time-set Θ constructed of temporal primitives that will be used throughout the rest of this work. Assume that set Σ contains all possible temporal primitives of one kind that is adopted by the ontology. Then the time-set Θ is defined as a subset of a power-set of Σ:

$$\Theta \subseteq P(\Sigma).$$

The empty set $\varnothing$ is also a member of Θ. For the rest of this work, we will use time-set Θ to represent temporal primitives that are referenced. In each particular case, a specific primitive will be selected (for example, points or intervals).

Set Θ is a dense unbounded linear set of temporal primitives. Set Θ has the following properties:

- Disjunction $\cup$:

  for each $\theta_1, \theta_2 \in \Theta, \ \theta_1 \cup \theta_2 = \theta_3, \theta_3 \in \Theta$;

- Conjunction $\cap$:

  for each $\theta_1, \theta_2 \in \Theta, \ \theta_1 \cap \theta_2 = \theta_3, \theta_3 \in \Theta$;

- Combination/composition $\oplus$:

  for each $\theta_1, \theta_2 \in \Theta$, exists such $\theta_3 \in \Theta$, that $\theta_1 \oplus \theta_2 = \theta_3$;

- Identity element $\varnothing$ (empty set):

  there exists such $\varnothing \in \Theta$, that for any $\theta \in \Theta, \ \theta \oplus \varnothing = \theta$.

Combination operation is very important for the temporal propagation. Starting time of a given event is combined with the duration of this event to obtain the ending time. Both the starting time and the duration of the event are represented using the values that are members of the time-set Θ. These values are combined using the combination operation $\oplus$.

Let us look into how to combine or to obtain a transitive closure of a pair of temporal constraints so as to get the effective constraints of both. Suppose that two events, namely e_1 and e_2, take place so that e_2 immediately follows e_1. Assume that e_1 starts and ends at t_1 and t_2 respectively. Since e_2 starts at the end of e_1, e_2 must start at t_2. Suppose that e_2 ends at t_3. Also assume that none of the events has a fixed duration, instead their durations are bounded. For example, the upper and the lower bound of an event e_k is respectively d^u_k and d^l_k. By rewriting the end point relationship for the events, we have the following constraints:

$$d^l_1 \leq t_2 - t_1 \leq d^u_1 \tag{1}$$

$$d^l_2 \leq t_3 - t_2 \leq d^u_2 \tag{2}$$

Adding equations 1 and 2 $\quad d^l_1 + d^l_2 \leq t_3 - t_1 \leq d^u_1 + d^u_2$

The resulting expression is known as temporal composition of metric bounds and is denoted as $[d^l_1, d^u_1] \oplus [d^l_2, d^u_2] = [d^l_1 + d^l_2, \ d^u_1 + d^u_2]$. The idea of temporal composition of metric bounds [9] is extended to temporal composition of durations with possibilistic distributions.

Suppose that possibilistic distributions of the temporal distance from t_1 to t_2, and the temporal distance from t_2 and t_3 are respectively $f(x)$ and $g(y)$. Possibilistic distribution of temporal distance from t_1 to t_3, say $F(z)$, can be computed from $f(x)$ and $g(y)$. Let z_1 be $x_1 + y_1$. A combination of possibilistic values $f(x_1)$ and $g(y_1)$ is $\min(f(x_1), g(y_1))$ for $z_1 = x_1 + y_1$. Taking all the possible combinations for z_1 as a summation of x and y, we get $F(z_1)$ as $\max(\min(f(x_k), g(y_k)| z_1 = x_k + y_k)$. This can be generalized as

$$F(z) = f(x) \oplus g(y) = \sup(\min(f_i(x), g_j(y)) \mid x + y = z).$$

Consider two functions, modeled by their respective possibilistic distributions

$$f(x) = [\{f_i(x) = \{x_i(\pi(x_i))\}, \forall\, i = 0,\ldots n\}]$$

and

$$g(y) = [\{g_j(y) = \{y_j(\pi(y_j))\}, \forall\, j = 0,\ldots k\}].$$

According to the results obtained in [11] and [22], functions $f(x)$ and $g(y)$ can be combined together to form combination $f(x)\oplus g(y)$. Such a combination is a double union of all possible pairs of each elementary distribution from $f(x)$ with each elementary distribution from $g(y)$:

$$(f \oplus g)(x) = \bigcup_{i=0}^{k}\bigcup_{j=0}^{n} f_i(x) \oplus g_j(y) =$$

$$= \bigcup_{i=0}^{k}\bigcup_{j=0}^{n} [x_i(\pi(x_i)), x_{i+1}(\pi(x_{i+1}))] \oplus [y_j(\pi(y_j)), y_{j+1}(\pi(y_{j+1}))].$$

The key element in the expression above is a combination $f_i(x) \oplus g_j(y)$ of two elementary trapezoidal distributions $f_i(x)$ and $g_j(y)$. Methods developed for trapezoidal distributions of Dubois and Prade are not applicable here because our trapezoids have different geometrical properties. A closed form formula subsuming all possible combinations of trapezoids (expressed by the relationships between parameters h_1, h_2, h_3 and h_4 of two trapezoids) is not known.

Depending on the interrelationship among the height parameters h_1 through h_4, the resulting formula describes a piece-wise linear function resulting from a combination of $f_i(x)$ and $g_j(y)$. The specific parameters of such a formula depend only on the analytical expressions for $f_i(x)$ and $g_j(y)$ and on up to two additional parameters x_1 and x_2, which can be derived from the analytical expressions of $f_i(x)$ and $g_j(y)$. The compositions of possible distributions are L to L, L to R, R to R and R to L, where L and R respectively denote L- and R-trapezoidal distributions. Consider L to L compositions of (a, b, h_1, h_2) and (c, d, h_3, h_4). By definition, $h_1 < h_2$ and $h_3 < h_4$. There are six possible cases given by the relationship among the heights of the trapezoid:

(1) $h_2 > h_1 \geq h_4 > h_3$

(2) $h_2 \geq h_4 > h_1 \geq h_3$

(3) $h_2 \geq h_4 > h_3 \geq h_1$

(4) $h_4 \geq h_2 > h_3 \geq h_1$

(5) $h_4 > h_3 \geq h_2 \geq h_1$ and

(6) $h_4 \geq h_2 > h_1 \geq h_3$.

An example of L to L composition that satisfies the conditions of case 2 is given below.

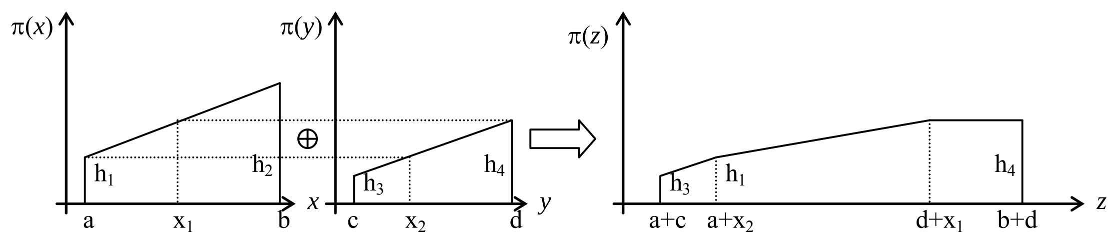

Figure 6. Example of composition of two L-trapezoidal distributions.

For L to R compositions there are 4 different cases each resulting in different shapes and these cases are:

(1) $h_1 \geq h_3$

(2) $h_2 \geq h_3 > h_1$

(3) $h_3 \geq h_2 > h_4$ and

(4) $h_4 \geq h_2$.

Similarly, for R to L composition there are 4 cases, and for R to R compositions there are 6 cases that result in different shapes. Detailed studies and the techniques for deriving the analytical expressions for such combinations are presented in [22].

When $h_1 = h_2$, the trapezoid becomes a rectangle and we provide the composition of rectangle with right and left trapezoid. Consider a rectangular distribution, say $f(x)$ given by $\{c, d, h_4, h_4\}$, with an L-trapezoidal distribution, say $g(x)$ given by $\{a, b, h_1, h_2\}$, where $h_1 < h_4 < h_2$. The combination of $f(x)$ and $g(x)$ can be expressed as

$$f(x) \oplus g(x) = g(x) \oplus f(x) = \{a+c, c+x_1, h_1, h_4\} \cup \{c+x_1, b+d, h_4, h_4\}$$

The results of this composition are shown in Figure 7.

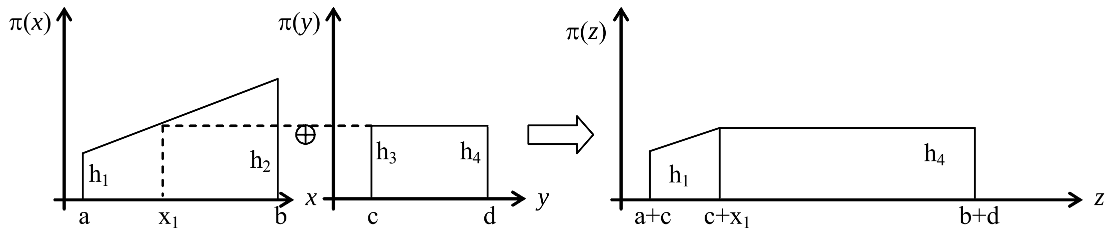

Figure 7. Combination of an L-trapezoidal and a rectangular distribution.

Figure 8 shows the results of combining an R-trapezoidal and a rectangular distribution.

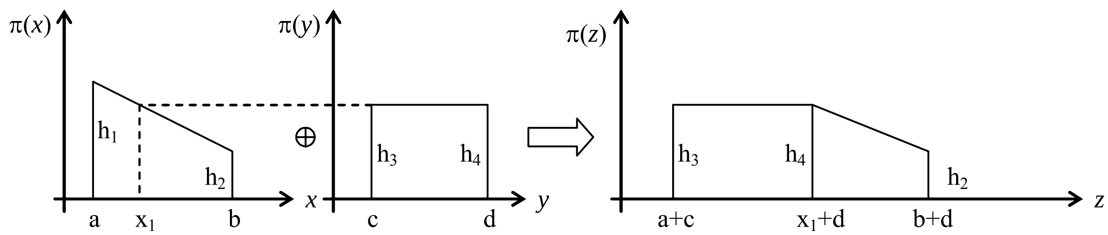

Figure 8. Combination of an R-trapezoidal and a rectangular distribution.

We can show that *composition of metric bounds is a special case of the composition of possibilistic distributions*. A metric bound $[d^l_k, d^u_k]$ is modeled as a possibilistic distribution $[1,1, d^l_k, d^u_k]$. The composition of a pair of metric bounds, say $[d^l_1, d^u_1]$ and $[d^l_2, d^u_2]$ is $[d^l_1 + d^l_2, d^u_1 + d^u_2]$. The corresponding composition of possibilistic distributions $[1,1, d^l_1, d^u_1]$ and $[1,1, d^l_2, d^u_2]$ is $[1, 1, d^l_1 + d^l_2, d^u_1 + d^u_2]$, which is the equivalent representation of the metric bound $[d^l_1 + d^l_2, d^u_1 + d^u_2]$. Hence, a composition of metric bounds is a special case of composition of possibilistic distributions.

5. Possibilistic Timed Petri nets

In the previous section, we have shown how an arbitrary possibilistic function can be approximated into a union of L- and R-trapezoidal distributions. We also outlined how to compose a pair of distributions. The presence of a token in a place represents the availability or meeting a precondition of a process that is represented by a transition. In our possibilistic timed Petri network, a token is associated with a time stamp, specifically, a possibility distribution of its availability. In a timed Petri network, each transition has a duration corresponding to the time taken by the process being modeled. Consider a precise timed Petri network, in which every token and transition is associated with a precise time and duration respectively. Assume that a process takes *d* units of time to perform; the transition corresponding to the process is associated with *d* units of time. Suppose a token arrives in a place with time stamp *t* and a transition with precise duration d fires at time t, the newly created token after the firing of the transition now has a time stamp $t+d$.

In a possibilistic timed Petri network, each token is associated with a possibilistic distribution time stamp, say $f(x)$, and each place is also associated with a possibilistic distribution of processing time, say $g(y)$. When the transition fires in a simple case with one place and an arc, the token created after the firing has a new time distribution given by $f(x) \oplus g(y)$. When firing of a transition depends on many tokens, having different time distributions, it is not straightforward how to compute when to fire a transition and how to compute the time stamp of the new tokens.

To address the firing with imprecise time stamp attached to each token, and when a possibilistic distribution is attached to transitions, we define the effective enabling time and relevant terms before we describe the algorithm.

The *enabling time* E_i of transition t_i in a timed Petri net is 0, if t_i has not fired since the start of the net's evolution, otherwise it is $low(E_i) \in \Theta$, where E_i is the last firing time of t_i and $low(E_i)$ is its lower bound value.

An *effective enabling time* EE_i of transition t_i in a timed Petri net is $EE_i = max(\theta_{synch}, E_i)$, where

$$\theta_{synch} = \max_{p_j \in I(t_k)}\left(\min_{i \in p_j}(\theta_i)\right).$$

This definition specifies the effective enabling time of the transition t_j to be the later of its enabling time and the synchronized time of enabling tokens in places $p_i \in I(t_j)$. Enabling tokens are those tokens that have the earliest timestamp among all tokens in place p_i. The number of enabling tokens in place p_i is equal to $\#(p_i, I(t_j))$, i.e. the multiplicity of an input place p_i of transition t_j.

Transition $t_j \in T$ of a marked timed Petri net $M = <P, T, D, I, O, \theta>$ is *enabled* if for all $p_i \in P$

$$\mu(p_i) \geq \#(p_i, I(t_j)).$$

Transition $t_j \in T$ of a marked timed Petri net $M = <P, T, D, I, O, \theta>$ is a *fireable transition* if it enabled and it has the minimal effective enabling time among all enabled transitions. It is possible that there will be no fireable transitions (no transitions are enabled), exactly one fireable transition, or more than one fireable transition (when they all have the same effective enabling time).

Transition $t_j \in T$ in a marked Petri net $M = <P, T, I, O, \mu>$ may *fire* whenever it is fireable. Firing of a fireable transition t_j results in a new marking $\mu'(p_i)$:

- $\#(p_i, I(t_j))$ enabling tokens are removed from each input place $p_i \in I(t_j)$;
- Enabling time E_i of the firing transition t_j is updated: $E_i^{'} = low(EE_i + d_i)$, where $low(x)$ is the lower bound of x as defined in [10];
- $\#(p_i, O(t_j))$ new tokens timestamped $(EE_i + d_i)$ are added into each output place $p_i \in O(t_j)$.

Ordinary Petri nets have a non-deterministic property of selecting which transition to fire among all enabled transitions [27]. This results in an explosion of the number of possible states of a given Petri net. In timed Petri nets, the size of the state space is greatly reduced by adding temporal criterion in the selection of the firing transition. For example, consider a fragment of an ordinary Petri net shown in Figure 9. The initial marking includes one token in place p_2 and one token in place p_3. Transitions t_2 and t_3 are both enabled to fire. In the framework of ordinary Petri nets such situation is resolved by randomly breaking a tie.

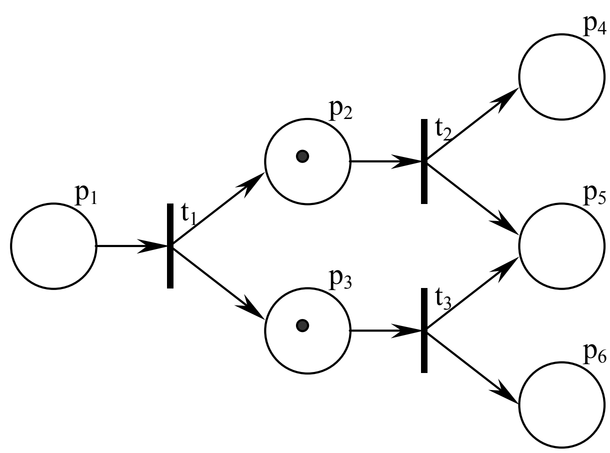

Figure 9. Non-deterministic property of a Petri net.

Consider a timed Petri net with the configuration and initial marking shown in Figure 9. Assume that token in place p_2 is timestamped with $\theta_2 \in \Theta$ and token in place p_3 has timestamp $\theta_3 \in \Theta$, and $\theta_2 < \theta_3$. Also assume that enabling time of transition t_2 is $E_2 \in \Theta$ and enabling time of transition t_3 is $E_3 \in \Theta$. Given this information and the configuration of the net, effective enabling time of transition t_2 is $EE_2 = \theta_2 \in \Theta$ and enabling time of transition t_3 is $EE_3 = \theta_3 \in \Theta$. Since $\theta_2 < \theta_3$, effective enabling time of transition t_2 is less than that of transition t_3, or $EE_2 < EE_3$. Therefore, transition t_2 has the minimal effective enabling time among all enabled transitions and is fireable according to the definitions above.

6. Algorithms for Timed Petri nets with possibilistic time

Any Petri network executes by firing transitions. In the classical Petri network model, as well as in our approach, at most one transition fires in each cycle. Figure 10 presents an algorithm of executing a Petri network. It runs in a loop until certain termination condition is met. It can be either a number of firings (cycles), a duration of the simulation, a condition when a token reaches certain place, etc. In each iteration of the loop, this algorithm calls procedure `FireTransition`, which finds an enabled transition and if such transition exists, fires it.

```
procedure PetriNet(PN, TC)
   input: Petri network PN;
   input: terminating condition TC;
   output: none;
begin
   while not TC do
      FireTransition(PN);
   End
end
```

Figure 10. The algorithm for execution of a Petri net

A transition may fire only if it is enabled, that is, each of its input places has at least as many tokens in it as arcs from the place to the transition. Figure 11 presents algorithm `FireTransition` for firing a transition, which starts with finding all enabled transitions in the network. Enabled transitions are found according to the rule given above, which is taken directly from the applied Petri network theory.

When many enabled transitions are available, the execution algorithm must determine which of them will fire. This is a source of non-determinism in the classical Petri net theory. In our approach we use temporal information stored in the tokens for determining which one among the enabled transitions will fire. First, *MinToken* with the earliest timestamp of the tokens in the places of all enabled transitions must be determined. The timestamps of these tokens will be used for the synchronization at the enabled transitions.

For each set of input places of each enabled transition we find a maximum (a synchronized token) among previously found tokens *MinToken*s. This shows the earliest time when tokens are available in all input places of an enabled transition and therefore, the time when a transition becomes enabled to fire. By comparing these times for all enabled transitions, we can find which is the earliest enabled one to fire. There are several ways of comparing possibilistic distributions. On of them is comparing the leftmost boundary of each distribution. Another one involves calculating a median value of each distribution and comparing these medians.

```
algorithm FireTransition(PN)
   input: Petri network PN
   output: Success or NoTransitionToFire
begin
   Token MinToken ← +Infinity
   Transition Firing ← ∅

   for all transitions T of PN do
      begin
         token MaxToken ← -1
         for all input places P of transition T do
            begin
               assume P is enabled
               if number of tokens in P < #(P, Input(T)) then
                   P is not enabled
               else if MaxToken < max(tokens in P) then
                   MaxToken ← max(tokens in P)
            end
         if P is enabled and MinToken > MaxToken and
                            MinToken > enabling time of T then
            begin
               Firing ← T
               MinToken ← max(MaxToken, enabling time of T)
            end
      end

   if Firing = ∅
     return NoTransitionToFire

   Token NewToken ← (MinToken + duration of Firing)
   update enabling time of Firing using NewToken

   for all input places P of Firing do
      remove minimal token from P

   for all output places P of Firing do
      add token NewToken into P

   return Success
end
```

Figure 11. The algorithm for firing a transition in a possibilistic timed Petri net.

Thus, among all enabled transitions, the firing transition is determined as the one whose synchronized token is minimal. If there are no enabled transitions (and therefore no firing transition), algorithm `FireTransition` returns a failure. Otherwise, the selected transition fires. From all input places of the firing transition a minimal token is removed. Adding the duration of the firing transition to the synchronized token generates a new token. This new token is added to all output places of the firing transition.

The algorithm for finding the firing a transition and firing it is extended as shown on Figure 11 to take into account that tokens are time stamped, transitions have durations, and they cannot fire again until a period of time passes that is equal to the duration of firing.

To illustrate the advantages of using possibility for representing temporal constraints, let us consider a solution of the problem of modeling the automated manufacturing system described in Section 2. Assume that a possibilistic representation of the operations' durations is such as shown in Table 1.

Duration of operation	Possibilistic distribution
Machine 1	[0.5min(0), 1min(1), 2min(1)]
Machine 2	[2min(1), 4min(1), 7min(0)]
Machine 3	[2min(0), 3min(1), 5 min(1), 8min(0)]
Machine 4	[3min(0), 6min(1), 7min(1), 10min(0)]
Robot 1	[0.5min(0), 1min(1), 1.5min(0)]
Robot 2	[0.5min(0), 1min(1), 1.5min(0)]

Table 1. Numeric data for durations of operations of the automated manufacturing system problem

The graph of the corresponding Timed Petri net and the configuration of the initial marking are presented in Figure 12. Assume the numeric values of the initial marking as shown in Table 2 (Figure 12 shows only the locations of tokens, but not possibilistic distributions associated with them).

Token	Located in place	Possibilistic distribution of the token
τ_1	p_1	[0min(0), 1min(1), 2min(1), 3min(0)]
τ_2	p_1	[0min(0), 1min(1), 2min(1), 3min(0)]
τ_3	p_5	[0min(0), 2min(1), 4min(1), 6min(0)]

Table 2. Numeric data for the initial marking of the automated manufacturing system problem

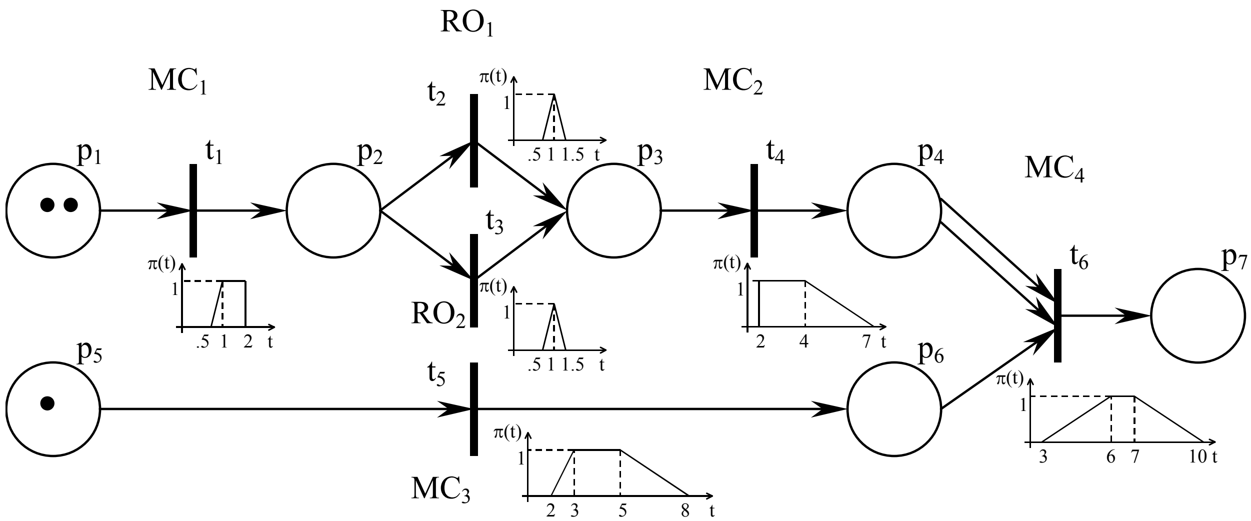

Figure 12. Timed Petri net graph for the automated manufacturing system

Tokens in place p_1 indicate that two units of part A will be available for machine 1 at any time between 0 and 3 minutes, but most plausibly between 1 and 2 minutes. Token in place p_5 indicates that machine 3 will receive part B at any time between 0 and 6 minutes, but most plausibly between 2 and 4 minutes.

Comparing the results with interval and possibilistic representation of time, it is obvious that the second result with possibilistic temporal intervals expresses much more information providing the most plausible interval for the final product to be assembled.

7. Summary and Conclusions

Petri networks have been a very useful for modeling processes for studying synchronization behaviors and properties such as deadlock and starvation. Time is implicitly represented and there is no notion of duration of a task that is being modeled, which hinders its utility for broad classes of problems, such as planning and scheduling, in which duration of tasks and time play a major role. A timed Petri network alleviates the problem by associating time stamps with tokens and time durations with transitions, places and arcs. We have provided an overview of how time is associated with a Petri network.

For many practical applications, duration of a task is not a fixed quantity. As a first approximation, a metric bound is being used to model duration. Some interesting work has been done using metric bounds [9] to represent tasks and reason with them. While an approach with metric bounds is being very useful, it fails to capture the processes accurately. Suppose, a process usually takes between 3 to 4 units and it occasionally takes 2 or 5 units of time. Using metric bounds, the process is modeled as a one that takes at least 2 units and at most 5 units. Thus, no distinction is made between the most likely scenario and the broad possible spread of the duration. To accurately model a process, we use a possibilistic distribution. We have shown a technique using a series of alternative R- and L- trapezoid distributions to approximate an arbitrary distribution. We have developed temporal composition operations between a pair of possibilistic distributions using which we can compute the completion time or the temporal distance between the starting and the ending time of the overall processes.

In a timed Petri network, we associate a possibilistic distribution with each token. Each transition models a task; hence it is associated with a possibilistic distribution of the task. We have illustrated the application of temporal composition of possibilistic distributions to compute time stamp of a token after a transition is fired. In a typical Petri network, synchronization or determining the enabling time of a transition is quite easy. However, it becomes quite challenging if a transition is associated with tokens with possibilistic distributions. We have developed a method to determining an effective enabling time of a transition with incoming tokens with possibilistic distributions. We have illustrated the utility of the proposed theory and method using an example of an automated manufacturing system. In summary, our approach is novel and has a broad utility beyond a timed Petri network and its applications.

References

[1] J. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26:832-843, 1983.

[2] I. Bestuzheva, V. Rudnev. Timed Petri Nets: Classification and Comparative Analysis. *Automation and Remote Control*, 51(10):1308-1318, Consultants Bureau, New York, 1990.

[3] C. Brown, D. Gurr. Temporal Logic and Categories of Petri Nets. In A. Lingass, R. Karlsson, editors, *Automata, Languages and Programming*, pp. 570-581, Springer-Verlag, New York, 1993.

[4] J. Carlier, P. Chretienne. Timed Petri Net Schedules. In G. Rozenberg, editor, *Advances in Petri Nets*, pp. 642-666, Springer-Verlag, New York, 1988.

[5] J. Cardoso, H. Camargo, editors. *Fuzziness in Petri Nets*, Physica Verlag, New York, NY, 1999.

[6] J. Cardoso, R. Valette, D. Dubois. Fuzzy Petri Nets: An Overview. In G. Rosenberg, editor, Proceedings of the *13th IFAC World Congress*, pp. 443-448, San Francisco CA, 30 June - 5 July 1996.

[7] J. Cardoso, R. Valette, D. Dubois. Possibilistic Petri nets. *IEEE transactions on Systems, Man and Cybernetics*, part B: Cybernetics, October 1999, Vol. 29, N 5, p. 573-582

[8] J. Coolahan. N. Roussopoulos. Timing Requirements for Time-driven Systems Using Augmented Petri Nets. *IEEE Transactions on Software Engineering*, 9(5):603-616, 1983.

[9] R. Dechter, I. Meiri, J. Pearl. Temporal Constraint Networks. *Artificial Intelligence*, 49:61-95, 1991.

[10] M. Diaz, P. Senac. Time Stream Petri Nets: a Model for Timed Multimedia Information. In R. Valette, editor, *Application and Theory of Petri Nets-94*, pp. 219-238, Springer-Verlag, New York, 1994.

[11] D. Dubois, H. Prade. *Possibility Theory*. Plenum Press, New York, 1988.

[12] D. Dubois, H. Prade. Processing Fuzzy Temporal Knowledge. *IEEE Transactions on Systems, Man and Cybernetics*, 19(4), July/August 1989.

[13] D. Dubois, J. Lang, H. Prade. Timed Possibilistic Logic. *Fundamenta Informaticae*. 15(3,4):211-234, 1991.

[14] D. Dubois, H. Prade. Processing Fuzzy Temporal Knowledge. *IEEE Transactions on Systems, Man and Cybernetics*, 19(4):729-744, 1989.

[15] M. Felder, A. Morzenti. A Temporal Logic Approach to Implementation and Refinement of Timed Petri Nets. In D. Gabbay, editor, Proceedings of *1st international conference on Temporal Logic ICTL-94*, Bonn, Germany, July 11-14, pp. 365-381, Springer-Verlag, New York, 1994.

[16] P. Fortemps. Jobshop Scheduling with Imprecise Durations: A Fuzzy Approach. *IEEE Transactions on Fuzzy Systems*, 5(4):557-569, 1997.

[17] C. Ghezzi, D. Mandrioli, S. Morasca, P. Mauro. A General Way to Put Time into Petri Nets. *ACM SIGSOFT Engineering Notes*, 14(3)-60-67, Pittsburgh, Pennsylvania, 1989.

[18] L. Godo, L. Vila. Possibilistic Temporal Reasoning Based on Fuzzy Temporal Constraints. In C. Mellish, editor, *Proceedings of IJCAI-95*, pp. 1916-1922, Montreal, Canada, 20-25 August, Morgan Kaufmann, San Francisco, CA, 1995.

[19] H. Hanisch. Analysis of Place/Transition Nets with Timed Arcs and Its Application to Batch Process Control. In M. Marsan, editor, *Application and Theory of Petri Nets-93*, pp. 282-299, Springer-Verlag, 1993.

[20] E. Kindler, T. Vesper. ESTL: A Temporal Logic for Events and States. In J. Desel, M. Silva, editors, *Application and Theory of Petri Nets-98*, pp. 365-384, Springer-Verlag, New York, 1998.

[21] L. Kunzle, R. Valette, B. Pradin-Chezalviel. Temporal Reasoning in Fuzzy Time Petri Nets. In J. Cardoso, H. Camargo, editors, *Fuzziness in Petri Nets*, pp. 146-173, Physica Verlag, New York, NY, 1999.

[22] S. Kurkovsky. Possibilistic Temporal Propagation. Ph.D. dissertation. University of Southwestern Louisiana, 1999.

[23] J. Lee, K. Liu, W. Chiang, Modeling Uncertainty Reasoning with Possibilistic Petri Nets. *IEEE Transactions on Systems, Man, and Cybernetics*. 33(2): 214-224, 2003.

[24] P. Merlin, D. Farber. Recoverability of Communication Protocols. *IEEE Transactions on Communications*, 24(9):541-580, 1989.

[25] D. Milutinovic, P. Lima: Petri Net Models of Robotic Tasks. Proceedings of the *2002 IEEE International Conference on Robotics and Automation*, Washington DC, vol.4, pp.4059-4064.

[26] C. Ramamoorthy, G. Ho, Performance Evaluation of Asynchronous Concurrent Systems Using Petri Nets. *IEEE Transactions on software Engineering*, 6(5):440-449, 1980.

[27] A. Raposo et al, Using Fuzzy Petri Nets to Coordinate Collaborative Activities. Proceedings of the *Joint 9th International Fuzzy Systems Association World Congress and 20th North American Fuzzy Information Processing Society International Conference*, p.1494 - 1499. Vancouver, Canada. IEEE Press, 2001.

[28] M. Tanabe. Timed Petri Nets and Temporal Linear Logic. In P. Azema, G. Balbo, editors, *Application and Theory of Petri Nets-97*, pp. 156-174, Springer-Verlag, New York, 1997.

[29] M. Woo, N. Qazi, A. Ghafoor. A Synchronization Framework for Communication of Pre-orchestrated Multimedia Information. *IEEE Networks*, 8(1)52-61, 1994.

[30] Y. Yao. A Petri Net Model for Temporal Knowledge Representation and Reasoning. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(9):1374-1382, 1994.