Computer Science Journal of Moldova, vol.**18**, no.1(52), 2010

Determining best-case and worst-case times of unknown paths in time workflow nets

Inga Camerzan

Abstract

In this paper we present a method aimed for determining best-case and worst-case times between two arbitrary states in a time workflow net. The method uses a discrete subset of the state space of the time workflow net and archives the results, which are integers.

1 Introduction

Time workflow nets (TWN) were developed to provide a suitable method to model, simulate, and analyze the behavior of time dependent systems, business processes. Best-case execution times (BCET) and worst-case execution times (WCET) are a necessary step in the development and validation process for hard real-time systems. Real-time systems need to satisfy stringent timing constraints, induced by the systems aims. We consider time workflow nets, those only which allow the modelling of time delay and deadlines for the execution of activities in the workflow process. This paper will mainly focus on modelling the control flow perspective. Thus the perspective workflow process definitions are formulated in order to specify which tasks need to be executed and in what order. If a worst-case input for the task were known, then there are reliable guarantees that processes always terminate. However the worst-case input is not known and it is hard to be determined.

In a workflow management system there is a delay beetween the moment when an activity becomes enabled and the moment when the activity is executed for a certain resource. For each transition t in the time workflow net there is a static interval $[a_t, b_t]$ associated to it. The times a_t, b_t are relative to the moment at which t was last enabled. Assuming that t was last enabled at the global time τ, then t may fire only during the interval $[a_t + \tau, b_t + \tau]$ and must fire the latest at the time $b_t + \tau$. This is a method of incorporating time into Petri Nets, introduced by Merlin [7] and studied in [1, 2, 8, 9, 10].

One of the most important problems, in the above mentioned nets, is to determine the deadlines for a sequence of system processes, i.e., to compare the longest duration of a transition sequence with a given limit. Such considerations are important for determining the best-case execution times and the worst-case execution times.

As a rule, the method used to estimate execution times bounds in practice consist in measuring the *end-to-end* execution time of the task for a subset of the possible executions, called test cases. This determines the minimal observed and the maximal observed execution times. Generally speaking, through these methods we are able to obtain an overestimation of the BCET and underestimation of the WCET, therefore we cannot consider them safe. Our aim is to propose an algorithm able to estimate the execution times in a safe way.

This paper is organized as follows: In Section 2 we introduce the basic notions and definitions for time workflow nets; In Section 3 we present the reachability graph of the time workflow nets, which can be used for computing the shortest and the longest path between two arbitrary states in the TWN; In the last section the algorithm for execution times estimation is proposed. The way it functions is illustrated by an example.

2 Basic notions and definitions

Definition 2.1 *A Petri net PN =(P, T, F, W) is a Workflow net iff:*

1. *PN has two additional places i and o, "start" place i, "destination" place o.*

2. *If we add a transition t^* to PN which connects o with i then the resulting Petri net is strongly connected.*

There are distinct methods of incorporating time in Petri nets: associating time delay to transition, associating time delay to places, associating time delay with arcs, associating time delays or time intervals to different types of objects of the net, associating stochastic time. Further we consider only Petri nets [5] which have deterministic time associated to transitions, in the form of time intervals, defined by Merlin [7] in 1972 and studied in [1, 2, 8, 9, 10].

We define a time workflow net in following way:

Definition 2.2 *A Time workflow net is a tuple Σ=(P, T, F, W, I) where PN=(P, T, F, W) is the workflow net (also called skeleton net), I: T$\rightarrow$ $Q_0^+ \times Q_0^+$ is a time function which associates timed intervals with transitions and $I_1(t) \leq I_2(t)$, where I(t) = $(I_1(t), I_2(t))$, for each transition t$\in$T.*

A global clock is associated with the time workflow net, which begins to work as soon as the first token appears in the net. After time association, the workflow net will work in the following way: from the moment when a transition t is enabled, the tokens from the input locations are stored for $I_2(t) - I_1(t)$ time units, and after this time elapses the transition fires putting tokens in their output places. For transitions in conflict, the first transition that fires is the one which has the latest time interval smaller.

For the definition of a state and of a change of state of a net we will follow [3, 6]:

Definition 2.3 *Let Σ=(P, T, F, W, I) be a time workflow net and J: T$\rightarrow$ $Q_0^+ \cup \{\sharp\}$. S=(m, J) is the state of the net Σ iff:*

1. *m is a marking in skeleton net,*
2. *if $t \in T$ and $t^- \leq m$, then $J(t) \leq I_2(t)$,*
3. *if $t \in T$ and $t^- \not\leq m$, then $J(t) = \sharp$,*

where $t^-(p) = W(p,t)$ is arc weight from place p to transition t.

We understand the notion of state in the following way. Let $S = (m, J)$ be a state. Each transition t in the net has a watch. The watch doesn't work ($J(t) = \sharp$) at the marking m if t is disabled at m. If t is enabled at m, then the watch of t shows the time $J(t)$ that has elapsed since t was last enabled.

Let $\Sigma = (P, T, F, W, I)$ be a time workflow net. The state $S_0 := (i, J_0)$ with i the initial marking of the time workflow net (the marking which has a single token in place i) and $J_0(t)= \begin{cases} 0, & \text{if} \quad t^- \le m, \\ \sharp, & \text{if} \quad t^- \not\le m \end{cases}$ is considered to be the initial state of the time workflow net. The states in a time workflow net can change due to transition firings or time elapsing.

Definition 2.4 *A transition t is enabled at the state S=(m, J), denoted by $S \to$, iff*

1. $t^- \le m$;
2. $I_1(t) \le J(t)$.

Thus, a transition is enabled in a time workflow net Σ, if t is enabled in the skeleton net (the timeless net) and the time specifications are satisfied, i.e t has been enabled for a sufficient amount of time. The resulting state is defined as follows:

Definition 2.5 *A transition t enabled at the state S=(m, J), will fire inducing state $S' = (m', J')$, denoted by $S \to S'$ defined thus:*

1. $m\prime = m + \triangle t$;
2. $$J'(t) = \begin{cases} \sharp, & t^- \not\le m', \\ J(t), & t^- \le m \wedge t^- \le m\prime \wedge F_t \cap F'_t = 0, \\ 0, & \textit{otherwise}, \end{cases}$$

   where $\triangle t = W(t,p) - W(p,t)$.

The resulting state (m', J') has a marking m' which results by firing of transition t in the skeleton net and a time vector J'. The values of the vector for the not enabled transitions in the marking are undefined $\sharp$. If a transition was enabled in the old marking, and it is still enabled in the new marking, and it is not in conflict with the just fired transition, then it keeps the values of its local clock $J'(t) = J(t)$. Otherwise, if a transition has just become enabled in the new marking, then its local clock $J'(t) = 0$.

Definition 2.6 *Let $\Sigma = (P, T, F, W, I)$ be a time workflow net. The state $S = (m, J)$ changes into the state $S' = (m', J')$ by the time duration $\tau \in Q$, denoted by $S \xrightarrow{\tau} S'$ iff $m' = m$ and the time duration τ is possible i.e. for any $t \in T$ with $J(t) \neq \sharp$, we have $J(t) + \tau \leq I_2(t))$ and*
$$J'(t) = \begin{cases} J(t) + \tau, & \text{if} \quad t^- \leq m, \\ \sharp, & \text{if} \quad t^- \not\leq m. \end{cases}$$

The sequence of transitions and time durations $\sigma = \tau_0, t_0, \tau_1, t_1, \ldots \tau_{n-1}, t_{n-1}$ is executable in the net Σ iff there exist the states $S_0, S'_0, S_1, S'_1, \ldots S'_{n-1}, S_n$ so that: $S_0 \xrightarrow{\tau_0} S'_0 \xrightarrow{t_0} S_1, \ldots S_{n-1} \xrightarrow{\tau_{n-1}} S'_{n-1} \xrightarrow{t_{n-1}} S_n$. That sequence shortly can also be noted by $S_0[\sigma\rangle S_n$. The transition-time sequence σ is called an execution sequence in the net Σ. State S' is reachable from the state S if there is a transition-time sequence σ so that $S[\sigma\rangle S'$. $RS(\Sigma, S_0)$ or $[S_0\rangle$ denotes the set of all reachable states of a net Σ and $R_\Sigma(S)$ denotes the set of all reachable states from the state S.

In Figure 1, the initial state is ((1,0,0,0,0,0)(0,$\sharp$, $\sharp$, $\sharp$)). Marking (1,0,0,0,0,0) is the initial marking of the skeleton net. The time vector has value 0 corresponding to transition t_1 enabled in the initial marking and the values $\sharp$ for the rest of the transitions which are disabled. Since t_1 is enabled at marking i and the time constraints are also satisfied, transition t_1 can fire in the initial state of the net Σ. The resulting state is $S_0 \xrightarrow{t_1} S_1 = ((0, 1, 1, 0, 0, 0)(\sharp, 0, 0, \sharp))$. The state S_1 has the marking $m_1 = (0, 1, 1, 0, 0, 0)$ and the time vector $J_1 = (\sharp, 0, 0, \sharp)$ with value 0 corresponding to the transitions t_2 and t_3, enabled in the marking m_1 in the skeleton net, and value $\sharp$ corresponding to the disabled transitions in the skeleton net. In the case of the time constraints, i.e. in the conditions $I_1(t_2) \geq J_1(t_2)$ and $I_1(t_3) \geq J_1(t_3)$ in state S_1 the transitions t_2 and t_3 cannot fire. So, state S_1 can change into another state in the net Σ only by time elapsing. For instance, the following state change is possible in the net Σ: $S_1 \overset{3.7}{\rightarrow} S_2$, where $S_2 = ((0,1,1,0,0,0)(\sharp, 3.7, 3.7, \sharp))$. We notice that the marking of the skeleton net remains unchanged, and results in a new time vector J_2 with updated time values for the transitions t_2 and t_3. The time duration 6.1 is not possible in state S_1 because the transition t_2, which, if enabled, must fire in 5 units of time. Now, both t_2 and t_3 can fire at state S_2. If t_2 fires, then we have $S_2 \overset{t_2}{\rightarrow} S_3 = ((0,0,1,1,0,0)(\sharp, \sharp, 3.7, \sharp))$. Now, the time duration 1.3 is possible at state S_3 and a new state appears: $S_3 \overset{1.3}{\rightarrow} S_4 = ((0,0,1,1,0,0)(\sharp, \sharp, 5, \sharp))$. Thus the state-transition-times sequence results in: $S_0 \overset{t_1}{\rightarrow} S_1 \overset{3.7}{\rightarrow} S_2 \overset{t_2}{\rightarrow} S_3 \overset{1.3}{\rightarrow} S_4$. The sequence $t_1, 3.7, t_2, 1.3$ is an execution sequence in the net Σ.

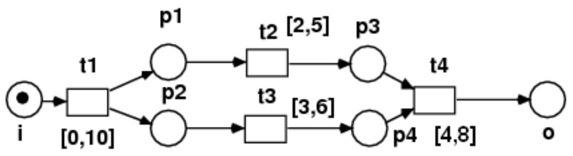

Figure 1. A time workflow net

3 Reachability graph of the time workflow net

The state space of a time workflow net is the set of all reachable states of the skeleton net, starting from i. Because of the markings, reachable in the net, this set is discrete, and it may be infinite. On the other hand, this set may be infinite because of the time of transitions. Thus, the set of all reachable states for a fix marking is infinite (and densely ordered) in general. Nevertheless, it is possible to pick up some "essential" states only, so that quantitative and qualitative analysis is possible. In [10] is shown, that essential states are integer states.

Definition 3.1 *The graph $RG(\Sigma, i)$ is called the reachability graph of the TWN Σ from initial state i iff its vertices are the reachable integer-states and its edges are defined by the triples (S, t, S') and (S, τ, S'), where $S \xrightarrow{t} S'$ or $S \xrightarrow{\tau} S'$, respectively.*

The reachable integer-states are those states from the state space, which have clocks that are (of enabled transitions) integers only and can be reached from the initial state S_0 through any number of transition firings or time durations. The reachability graph of the time workflow net Σ is the transition relation $\rightarrow$ restricted to its reachable integer-states. This graph is finite iff the set of the reachable markings of the time workflow net is finite. This set is finite, if the set of reachable markings of the skeleton net is finite.

Using the parametric description of transition sequence [7] minimal and maximal length of time of the execution sequence can be evaluated. The maximal and minimal length of time is an integer and it can be reached by firing in integer-states. Thus, when the TWN is bounded, a sequence with maximal/minimal length of time can be found for given source-state and sink-state.

For time workflow net above the reachability graph consists of following reachable integer-states:

$m_0 = (1,0,0,0,0,0)$, $J_0 = (0,\sharp,\sharp,\sharp)^T$, $J_0^* = (10,\sharp,\sharp,\sharp)^T$,

$m_1 = (0,1,1,0,0,0)$, $J_1 = (\sharp,0,0,\sharp)^T$, $J_1^* = (\sharp,2,2,\sharp)^T$, $J_1^{**} = (\sharp,5,5,\sharp)^T$,

$m_2 = (0,0,1,1,0,0)$, $J_2 = (\sharp,\sharp,2,\sharp)^T$, $J_2^* = (\sharp,\sharp,3,\sharp)^T$, $J_2^{**} = (\sharp,\sharp,6,\sharp)^T$,

$m_3 = (0,0,0,1,1,0)$, $J_3 = (\sharp,\sharp,\sharp,0)^T$, $J_3^* = (\sharp,\sharp,\sharp,4)^T$, $J_3^{**} = (\sharp,\sharp,\sharp,8)^T$,

$m_4 = (0,0,0,0,0,1)$, $J_4 = (\sharp,\sharp,\sharp,\sharp)^T$,

$S_8 = (m_0, J_0^*)$, $S_9 = (m_1, J_1^{**})$, $S_{10} = (m_2, J_2^{**})$, $S_{11} = (m_3, J_3^{**})$,

$$\underbrace{(m_0, J_0)}_{S_0} \xrightarrow{t_1} \underbrace{(m_1, J_1)}_{S_1} \xrightarrow{2} \underbrace{(m_1, J_1^*)}_{S_2} \xrightarrow{t_2} \underbrace{(m_2, J_2)}_{S_3} \xrightarrow{1} \underbrace{(m_2, J_2^*)}_{S_4} \xrightarrow{t_3}$$

$$\underbrace{(m_3, J_3)}_{S_5} \xrightarrow{4} \underbrace{(m_3, J_3^*)}_{S_6} \xrightarrow{t_4} \underbrace{(m_4, J_4)}_{S_7}$$

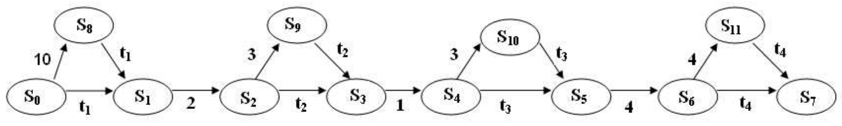

Figure 2. Reachability graph

4 Determining BCET and WCET

Methods from graph theory may be applied to determine best-case and worst-case execution times while constructing the reachability graph.

Determining the best-case execution leads directly to the well-known problem of the shortest path. Since reachability graph has nonnegative times only, all common shortest path algorithms are applicable, e.g. Dijkstra's algorithm or Bellman-Ford algorithm [4].

Determining worst-case execution is similar to the critical path problem, sometimes called longest path problem.

The problem can be formulated as follows:

For a given directed weighted graph $RG = (V, E)$, find the lengh l of a longest path from a source vertex v_s to a goal vertex v_d so that v_d is contained by most ones as a last vertex.

Actually we mean, that the length l is infinite, if there exists a cycle reachable starting on v_s before passing v_d and, otherwise, l is the sum of the weights of the longest path.

To determine worst-case execution, we propose the following algorithm A_1:

1. Remove from the graph RG all edges (v_d, v_j), i.e. all edges that are directed from v_d.

2. For each edge $(v_i, v_j) \in RG$ with the weight w_i assign a new weight $w_i^- = -w_i$. Edges, labeled by transitions names, obtain the weight 0.

3. Procedure Bellman-Ford (V, E, s) ** s is a source node

```
for each $v \in V(RG)$ do
        $d[v] \longleftarrow \infty$
        $p[v] \longleftarrow NIL$              ** $p[v]$ is predecessor node of v
$d[s] \longleftarrow 0$
for $i \longleftarrow 1$ to $n-1$               ** $n = |V(RG)|$
        for each edge $(u, v) \in E(RG)$ do
                if $d[u] + w(u, v) < d[v]$ then
                        $p[v] \longleftarrow u$
                        $d[v] \longleftarrow d[u] + w(u, v)$
for each edge $(u, v) \in E(RG)$ do
        if $d[u] + w(u, v) < d[v]$ then ** check for negative weight
cycles
                return $FALSE$
return $TRUE$
```

If algorithm returns false, then l is infinite. Otherwise, $l = -d(v_d)$. The complexity of this algorithm is dominated by the complexity of the Bellman-Ford algorithm, i.e. it is $O(|V| \cdot |E|)$. A correctness of A_1 is easy to be seen after the removal of all output edges from goal vertex. No path is possible, which contains goal vertex at another position than a final vertex. Obviously, the shortest path in the negative weighted graph corresponds to the longest path in the initial graph.

We computed a worst-case and a best-case execution times with the help of INA tool [11] for the example from the Figure 1. Our algorithm identified 11 states. The worst-case execution time of the service from source node i to target node o is 24 units of time. A maximal path is $i \Longrightarrow p_2 \Longrightarrow p_4 \Longrightarrow o$. The best-case execution time of service from source node i to target node o is 6 units of time. A minimal path is $i \Longrightarrow p_1 \Longrightarrow p_3 \Longrightarrow o$.

5 Conclusions

In this paper we presented a new approach aimed at determining best-case and worst-case times between two states in a TWN in polynomial time and demonstrated the application of our method for a certain time workflow net.

References

[1] B. Berthomieu, *Modeling and Verification of Time Dependent Systems Using Time Petri Nets*, In Advances of petri Nets 1984, vol 17 , No 3 of IEEE Trans. On Software Eng. 1991, 259–273.

[2] B. Berthomieu, *An Enumerative Approach for Analyzing Time*, In Proceedings IFIP 1983, R:E:A:Mason(ed), North-Holland, 1983, 41–47.

[3] I. Camerzan, *On soundness for time workflow nets*, Computer Sciece Journal of Moldova, volume 15, nr. 1(43), Chisinau, 2007, 74–87.

[4] T.H. Cormen, C.E. Leisserson, R. L. Rivest, C. Stein, *Introduction to Algorithms*, second edn. MIT Press, 2001.

[5] T. Jucan, F. Tiplea, *Petri Nets – theory and practice*, Academia Romana, Bucuresti, 1999. (in Romanian)

[6] T. Jucan, O. Prisecaru, I. Camerzan, *Time Interval Workflow Nets*, Scientific Annals of the "Al. I. Cuza" University, Computer Science Section, Tome XV, Iasi, 2005, 77–92.

[7] P. Merlin, *A study of the recoverability of computer system*, Ph. D. thesis, Dep. Computing Science, University California, Irvine, 1974.

[8] L. Popova-Zeugmann, *On Time Invariance in Time Petri Nets*, In Informatik-Bericht Nr. 36 der Institute pur Informatik der Humboldt-Univ. Zu Berlin, Oct. 1994

[9] L. Popova-Zeugmann, *On Liveness and Boundness in Time Petri Nets*

[10] L. Popova-Zeugmann, *On Parametrical Sequences in Time Petri Nets*, Proceedings of the CSP 97 Workshop, Warsaw(1997), 105–111.

[11] P.H. Starke, *INA – Integrated Net Analyzer*, Berlin, 1997.

Inga Camerzan,

Received March 4, 2010

State University of Tiraspol
E–mail: *caminga*2002@*yahoo.com*