Improving the Efficiency of Workflow Analysis

Loucif. Zerguini

Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O. Box 514, NL-5600 MB, Eindhoven, The Netherlands
E-mail: L.Zerguini@tue.nl

Abstract. This paper presents an approach for the improvement of the efficiency of the performance analysis of large workflow models. We propose a simple and powerful decidable Petri net reduction technique that reduce the size of a workflow model. Our method avoids the direct solution of the original model.

1 Introduction

An often used quantitative measure to assess the efficiency of a business process is its response time [3]. Business analysts need quick estimates of the response time distribution to check whether a design satisfies user requirements. For the purpose of performance analysis, stochastic delays are more expressive than fixed or interval delays. Many researches have been done under a workflow definition built by conventional workflow routing constructs like sequential routing , parallel routing , choice routing and iterative routing ([3], [6]). Moreover, there are numerous techniques for deriving performance measures [5]. Despite these results, the state-space explosion problem remains the major difficulty for using these methods in practical applications. In this paper, we propose a methodology avoiding the direct solution of the original model which consists of evaluating simpler models to derive performance measures on the complex underlying model with general topology. Our goal is to provide business process designers with a powerful tool to improve the efficiency of the analysis of large workflow models.

2 Stochastic Modeling of Workflows

A Petri net (PN) is a triple *(P, T, F)* where *P* is a finite set of places, *T* is a finite set of transitions ($P \cap T = \emptyset$) and $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (*flow relation)* (see [4] for a complete description). A Workflow net (WF-net) is a subclass of Petri nets tailored toward workflow analysis . It has a unique source place i and unique sink place o . In such a net a task is modeled by a transition and intermediate states are modeled by places. A token in the source place i corresponds to a case (workflow instance) which needs to be handled, a token in sink place o corresponds to a case that has been handled. The process state is defined by a marking. In addition, a WF-net requires all nodes (i.e. transitions and places) to be on some path from i to o. This ensures that every task (transition) and every condition (place) contributes to the processing of cases. A WF-net is safe iff for each place p and for every reachable state the maximum number of tokens in p does not exceed 1. A WF-net is sound if a process can always terminate with a single token in place o and all other places are empty and there is no dead task, i.e. each task can be executed (see [1] for a complete description of all notions related to a WF-net). A set of input (resp. output) transitions of a place $p \in P$ is denoted by ${}^\bullet p$ (resp. $p^\bullet$) and the set of input (resp.output) places of a transition $t \in T$ is denoted by ${}^\bullet t$ (resp. $t^\bullet$); for $X \subseteq (P \cup T)$, ${}^\bullet X = \bigcup_{x \in X} {}^\bullet x$ and $X^\bullet = \bigcup_{x \in X} x^\bullet$. Let $N = (P, T, F)$ and $N_0 = (P_0, T_0, F_0)$ be WF- nets. N_0 is a subflow of N iff $P_0 \subseteq P$, $T_0 \subseteq T$, and $F_0 = F \cap ((P_0 \times T_0) \cup (T_0 \times P_0))$.

A stochastic workflow net (SWF-net) is a tuple *(P, T, F, D,W)* where

1. (P, T, F) is a sound and safe WF-net,
2. D: $T \longrightarrow (I\!R^+ \longrightarrow [0, 1])$ is the probability distribution function for the firing time of $t \in T$,
3. $W : T \longrightarrow (I\!R^+ - \{0\})$ is the weight function.

The delay function D will be used to sample transition delays that represent the firing time of actual transition execution. A transition $t \in T$ is called $timed$ if $D_t(0) < 1$. A transition $t \in T$ is called $immediate$ if $D_t(0) = 1$. Pictorially, square boxes represent stochastic timed transitions and thin bars represent immediate transitions. An enabled transition can fire after time delay sampled from a delay function D. The weight function W is added to each transition to resolve conflicts, we denote the weight $W(t)$ of a transition $t \in T$ with $w(t)$. We assume implicitly $preselection\ policy$, this in contrast to the race semantics used in [5]. For example, in Fig.1, transition T_1 will fire with probability $\frac{w(T_1)}{w(T_1)+w(T_2)}$ and transition T_2 will fire with probability $\frac{w(T_2)}{w(T_1)+w(T_2)}$.

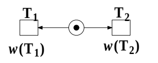

Fig. 1. Weight Function

The $preselection\ policy$ is reasonable in the context of workflow management, since tasks are often performed by human whose work typically cannot (or will not) be cancelled upon completion of other tasks. Except for the resolution of conflicts, we suppose in this paper that the semantic of SWF-net is the same as the semantic of the non-Markovian stochastic Petri nets defined in [5]. A SWF-net captures the dynamic behavior of a single case in isolation within a workflow. The stochastic process that it induces expresses the way that a specific case or workflow instance is handled. In this paper we focus on the time between the start and the end of the processing of a single case. Informally stated, this is the response time. In the sequel a sound and safe SWF-net will be simply denoted by (P, T, F).

3 Decomposing the SWF-net Model

An approach can be based on the analysis of the structure of the SWF-net model, with the aim of identifying embedded subflows, that are reducible. Reducible subflow (RSF) can be analyzed separately, and the result obtained in isolation can later be combined in order to produce the overall result.

3.1 Reducible Subflow

Definition 1. *Let G = (P, T, F) be a SWF-net and $X \subseteq P \cup T$ be a set of nodes, then a subflow N = ($P \cap X$, $T \cap X$, $F \cap (X \times X)$) is an RSF of G, if and only if $\exists\ Z_{in}$, $Z_{out} \in P$ such that:*
1. $|T \cap X| \geq 2$ (large enough, at least two transitions)
2. $^\bullet(X \backslash \{Z_{in}, Z_{out}\}) \subseteq (X \backslash \{Z_{out}\}) \wedge (X \backslash \{Z_{in}, Z_{out}\})^\bullet \subseteq (X \backslash \{Z_{in}\})$ (Z_{in} is the input place and Z_{out} is the output place).

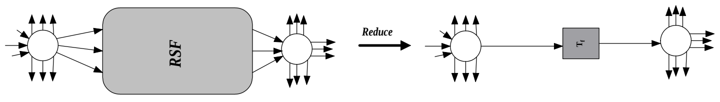

Fig. 2. RSF routing pattern performance equivalent analysis

In Figure 2, RSF denotes tasks in reducible RSF pattern, and transition T_f denotes the task that has the equivalent time performance to the RSF (one should note that this reduction is possible thanks to a soundness and a safeness of the corresponding net).

When any transition outside the RSF is enabled, the number of tokens in any place belonging to the $RSF - \{Z_{in}, Z_{out}\}$ remains invariable, the subflow *RSF* remains thus completely isolated from the rest of the SWF-net, and its evolution is independent from that of the rest of the model.We can thus analyze the stochastic behavior of each RSF in isolation and then combine the results, rather than analyzing the whole original system directly.

3.2 Algorithm

```
Algorithm 1: (Numbering Nodes Algorithm)
Input: x ∈ X
Output: $l(x)$ = level of $x$

Begin
    $X := \{i\}$
    $d := 0$
    $Y := \emptyset$
    while $X \neq \emptyset$ do
        for x ∈ X do
          $l(x) := d$
        od
         $Y$ := $Y$ ∪ $X$
         $X$ := $X^\bullet$ \ $Y$
         $d := d + 1$
    od
end
```

With the algorithm 1 we obtain a function *l* with which all the nodes of the given SWF-net can be numbered in strictly ascending order. This algorithm is related to those determining the level of the nodes in graphs (see [2]). The obtained function *l* will be used in lines 7 and 8 of the algorithm 2.

```
Algorithm 2: (RSF Detection Algorithm)

Input : G= $(P,T,F)$ a SWF-net
Output: RSF = reducible subflow

Begin
1.  $X := P$
2.   pick $a \in X$
3.   $Y := \{a\}$
4.   $Z_{in} := Z_{out} := \emptyset$
5.      while $^\bullet(Y\backslash\{Z_{in}, Z_{out}\}) \nsubseteq (Y\backslash\{Z_{out}\}) \vee (Y\backslash\{Z_{in}, Z_{out}\})^\bullet \nsubseteq (Y\backslash\{Z_{in}\}) \vee |T \cap X| < 2$
6.            $Y$ := $Y$ ∪ $^\bullet(Y\backslash\{Z_{in}, Z_{out}\})$ ∪ $(Y\backslash\{Z_{in}, Z_{out}\})^\bullet$
7.            $Z_{in} := \{x^* \in Y \cap P$ | $l(x^*) \leq l(x)$ for all x $\in Y \cap P\}$
8.            $Z_{out} := \{x^* \in Y \cap P$ | $l(x^*) \geq l(x)$ for all x $\in Y \cap P\}$
9.      od
10.     $Y$ is an $RSF$ of G with a source place $Z_{in}$ and a sink place $Z_{out}$
end
```

The straightforward algorithm 2 picks any place *a* from X, then it finds the embedded reducible subflow containing the place *a* by determining the source and the sink places using the numbering function given by the algorithm 1. By applying repeatedly this algorithm each of the reducible subflow can be decomposed into smaller reducible subflows until to obtain the skeletons contained in a given SWF-net. The net has $|P|$ places, $|T|$ transitions and $|F|$ arcs.

4 Application

Let's consider the reduction process depicted in the Fig.3. The first SWF-net has 8 places and 7 transitions.

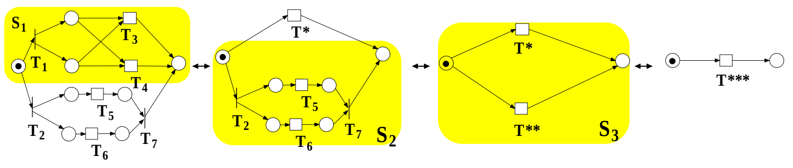

Fig. 3. Reduction Process

To determine the response time distribution of the whole system, we will proceed by decomposition as explained in the previous section.

– Step 1: Detect the first subflow S_1 (using algorithm 2) and analyse it. The equivalent time performance to the S_1 is represented by transition T^*. Clearly, the firing time distribution D_{T^*} of the equivalent transition T^* is equal to $\frac{w(T_3)}{w(T_3)+w(T_4)}$. D_{T_3} + $\frac{w(T_4)}{w(T_3)+w(T_4)}.D_{T_4}$.
– Step 2: Detect the second subflow S_2 (using algorithm 2) and analyse it. The equivalent time performance to the S_2 is represented by transition T^{**}. Clearly, the firing time distribution $D_{T^{**}}$ of the equivalent transition T^{**} is equal to the distribution of the maximum of two independents random variables. Thus, $D_{T^{**}}$ = $D_{T_5}.D_{T_6}$
– Step 3: Finally, the last subflow S_3 can easily be analyzed. The firing time distribution $D_{T^{***}}$ of the equivalent transition T^{***} which correspond here to the response time distribution of the whole system is equal to $\frac{w(T^*)}{w(T^*)+w(T^{**})}$. D_{T^*} + $\frac{w(T^{**})}{w(T^*)+w(T^{**})}.D_{T^{**}}$ (We should notice here that $w(T^*) = w(T_1)$ and $w(T^{**}) = w(T_2)$).

5 Conclusion

Due to the state-space explosion, a wide range of modelling problems concerning the evaluation of complex workflows, according to aspects like response time distribution, are very difficult to handle if they are not decomposed into separate submodels. Our decidable reductional approach provides an elegant, effective and efficient solution procedure by means of structural decomposition and aggregation. This approach is based on the identification of general reducible subflows. We have presented an automated technique to reduce the complexity to compute the response time distribution of SWF-net models with unrestricted topology. A workflow instance response time distribution can thus be analyzed more efficiently.

References

1. W.M.P.van.der Aalst. The application of Petri nets to workflow management. The Journal of Circuits, Systems and Computers, vol.8, pp. 21 – 66, 1998.
2. Berge, C. The theory of graphs and its applications. London- New York: 1964.
3. H. Jonkers, P. Boekhoudt, M. Rougoor and E. Wierstra. Response time and critical path analysis for the optimization of business process models, Proc. of the summer computer simulation symposium, pp. 222- 229, Chicago 1999.
4. T. Murata. Petri nets: properties, analysis and applications. Proc. IEEE, vol.77, pp. 541–580, April 1989.
5. A. Puliafito, M. Scarpa and K. S. Trivedi. Petri nets with k simultaneously enabled generally distributed timed transitions. Performance Evaluation, vol. 32, pp. 1–34, 1998.
6. L. Zerguini. Approximate computation of response time distribution in workflows. Proc. High Performance Computing Symposium (HPC'02), pp. 280–287, April 2002.