Extending a Response Time Approximation Technique to Colored Stochastic Petri Nets*

Jörn Freiheit and Armin Zimmermann
Performance Evaluation Group, Technische Universität Berlin
Franklinstr. 28/29, 10587 Berlin, Germany

August, 1998

Abstract

For the analysis of large systems modeled with stochastic Petri nets, state explosion is a well known problem. Many real-life systems are thus impossible to analyze. Several research activities try to overcome this limitation. Diverse approaches can be found in the literature, e.g. [1, 2, 3, 4].

This paper presents an iterative approximation technique for the steady-state throughput computation of complex concurrent systems. The proposed technique makes use of the divide and conquer principle. It is derived from the response time approximation method presented in [5, 6]. We generalize this approach to a special class [7] of hierarchical colored stochastic Petri nets [8].

1 Introduction

Verification of correctness and optimization are the main tasks during the planning and development of systems. Qualitative and quantitative analysis can be used for this. Petri nets are a well-known modeling means for the specification of systems. Moreover, the necessary analysis techniques are known from Petri net theory. The usual performance analysis methods require the generation of the whole state space of the model. For many systems of real-life size the state space is too large to be handled. This is called the state explosion problem and necessitates advanced techniques that overcome this limitation.

The throughput is often the most important performance measure of technical systems. An algorithm for the approximate analysis of GSPNs [9] has been presented in [5, 6]. However, colored Petri nets are better suited for the modeling of complex systems.

This paper describes the extension of the response time approximation method presented in [5, 6] to a special class of colored stochastic Petri nets. This modeling has been developed especialy for the description of manufacturing systems. They are a popular example for systems where the throughput is substantial for the design. The approximation technique presented in this paper is applied to a flexible manufacturing system example modeled using the mentioned class of colored Petri nets.

The method presented in this paper makes use of a model decomposition. Thus, several models with substantially smaller state space have to be analyzed. The disadvantage is that only approximate performance results are computed, although experiences show that the error is acceptable in most cases.

Decomposition methods commonly contain the following three steps: A partitioning of the analyzed system into smaller disjunctive subsystems has to be found first. This can be a serious problem for unstructured models like GSPNs. Dedicated colored Petri net models used here are hierarchically partitioned into *pages*. Those pages describe subsystems in a modular way. Thus, a simple method is to treat each page as a subsystem. Modules of real-life systems as well as modeled subsystems interact asynchronously by means of buffers. The application example presented in the paper shows that this clear and simple heuristics leads to acceptable results.

After the decomposition the main model is cut into submodels, which are not analyzeable in isolation. The task of the second step is then to build a so-called *low-level system* for each of the subsystems. This is done by keeping the current subsystem and adding an aggregation of the others. Additionally a *basic skeleton* is derived which includes an aggregation of all subsystems.

Approximate performance measures are computed during the third step. In this case the method presented here differs only slightly from the iterative re-

*This work has been supported by the German Research Council under grant Ho 1257/4-5

sponse time approximation technique developed in [5]. More details are given in section 4.

The main advantage of decomposition approaches is that the analysis of small subsystems needs less memory. Despite the need to iteratively repeat the algorithm, the approximate results are computed faster than with standard methods.

The remainder of this paper is organized as follows. The following section introduces the manufacturing system example that is used throughout the paper, and the proposed modeling method, which is applied to the example. In section 3 the partition and aggregation method is shown. The iterative approximation algorithm is the contents of the fourth section. Finally, section 5 provides some concluding remarks.

2 An Example Modeled With Dedicated Colored Petri Nets

This section describes a flexible manufacturing system example modeled with a dedicated colored Petri net. In this special Petri net class there are only two color types. Object tokens model workpieces inside the manufacturing system, and consist of a name and the piece's current state. Elementary tokens do not have a special color, and are equivalent to tokens from uncolored Petri nets. Elementary tokens are used to model the states of the machines, e.g. if they are failed or busy, or the position of a conveyor system. Places can contain either object tokens or elementary tokens. Object places are drawn as thick circles, and elementary places are drawn as thin circles, respectively. Transitions represent possible events in the system. Each input and output arc of a transition is connected to one place, and only tokens of the appropriate color type can flow through it. Therefore, arcs are drawn thick or thin as well, corresponding to their associated color type.

Each model consists of a structural model of the manufacturing system's layout and several work plan models of the production routes. The different model parts are automatically merged to create a complete model of the manufacturing system [7].

Model of the Structure

The structural model describes the resources and their work plan independent properties. Figure 1 shows the top layer of the hierarchical structural model of our manufacturing system example.

Each of the so called substitution transitions (depicted as ▯) is refined by a subpage that describes the behavior of a machine or a conveyor in more detail. Figure 2 shows the detailed structure of the assembly transition.

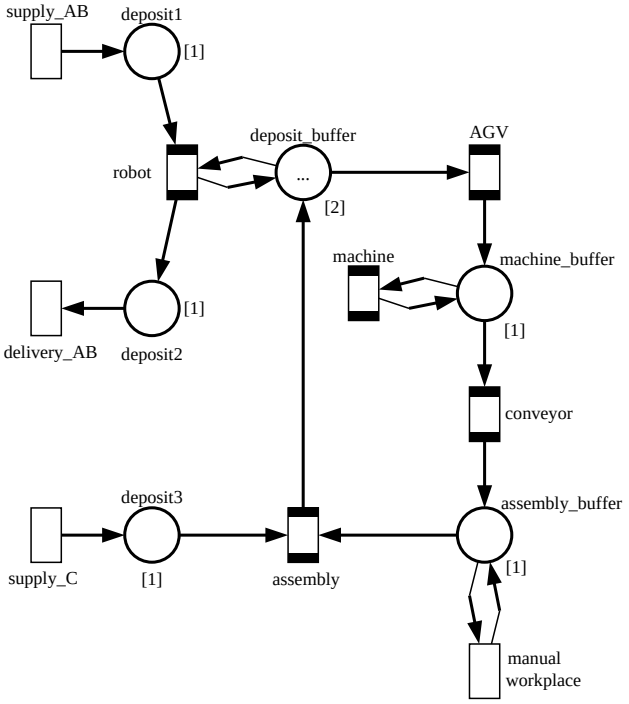

Figure 1: Manufacturing system example

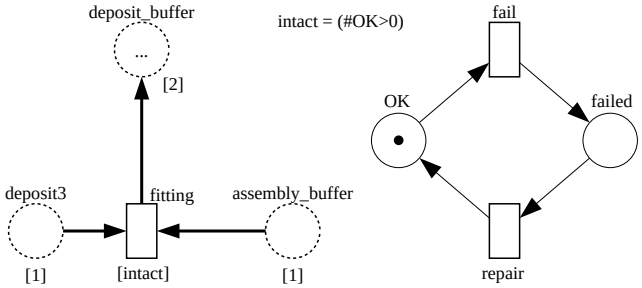

Figure 2: Detailed structure of the assembly part

Work Plan Models

The work plan models describe the production routes for each workpiece. Such a model represents paths through the structural model, hence only places and transitions that can be found there are usable here. This applies to all levels of hierarchy; the work plan models are refined in the same way as the structural model. Alternative routes of workpieces can be modeled using different paths, with conditions and probabilities assigned to them if needed. In our example we have three work plans; one for each of the workpieces A, B, and C.

3 Partition and Aggregation

In this section the low-level systems and a basic skeleton from the whole system are derived. The state

space of the low-level systems and the basic skeleton is usually smaller than the original one by more than one order of magnitude.

In each low-level system one subsystem is kept and the other subsystems are aggregated. Thus,as many low-level systems are derived as there are subsystems in the partitioned system. In the basic skeleton all subsystems are aggregated.

As described above, for each substitution transition (and thus each page) one subsystem is generated.

$N = (P, T, F)$ is a net if P and T are disjoint sets of places and transitions and $F \subseteq (P \times T) \cup (T \times P)$ is the set of arcs. The places connected to a subsystem are called *buffers*. The set of buffers is denoted with $B \subseteq P$ and the set of subsystems $S \subseteq N$ $(S_i = (P_i, T_i, F_i)$ with $S = S_1 \cup S_2 \cup \ldots \cup S_n$ where $S_1 \cap S_2 \cap \ldots \cap S_n = \emptyset$. The *preset* of B is denoted by ${}^\bullet B$, and the *postset* $B^\bullet$, respectively. The set of transitions $IT_i \subseteq T_i$ $(OT_i \subseteq T_i)$ where for each $t \in IT_i$ $(t \in OT_i)$ is $t \in B^\bullet$ $(t \in {}^\bullet B)$ is called *input (output) transitions*[1] *of the subsystem* S_i. There is a *path* between an input transition $t_1 \in IT_i$ and an output transition $t_2 \in OT_i$ if and only if $(t_1, t_2) \in F_i^*$, where F_i^* is the transitive closure of F_i.

Aggregation Rules for the Structural Model

Two aggregation rules for the structural model are given in the following. The first aggregation rule substitutes each path of the subsystem by a new place. If there is at least one path between two different transitions $t_1 \in IT_i$ and $t_2 \in OT_i$, delete the places and transitions in the path(s) and add a new place s with $(t_1, s) \cup F_i$ and $(s, t_2) \cup F_i$. The initial marking of the additional place is set to the sum of path's places. Figure 3 shows an example for the application of this aggregation rule.

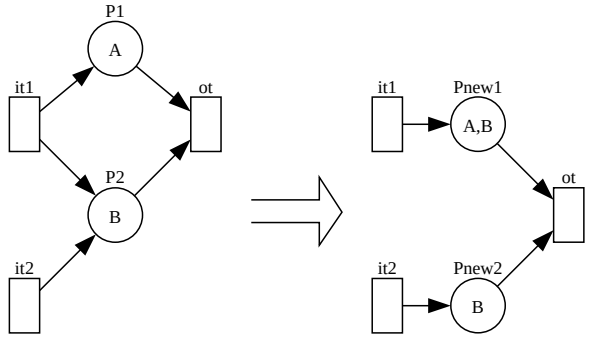

Figure 3: General aggregation rule

In the special case where a transition t is both input and output transition of a subsystem S_i $(t \in (IT_i \cap OT_i))$ a different aggregation rule is applied. Let $b, b' \in B$ be two buffers that are connected with t $((b,t) \in F$ and $(t,b') \in F)$ then add a new transition t_{new} and a place p_{new}. The set of arcs is adjusted as follows: $F_{new} = F \setminus \{(b,t)\} \cup \{(b, t_{new}), (t_{new}, p_{new}), (p_{new}, t)\}$. The initial marking of the additional place p_{new} is again set to the sum of path's places. Figure 4 shows the application of this special aggregation rule.

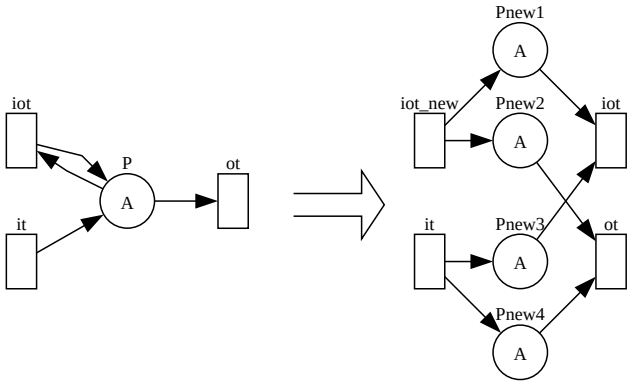

Figure 4: Special case aggregation rule

Aggregation Rules for the Work Plan Models

After generating the low-level subsystems and the basic skeleton for the structural model the same has to be done for each work plan model. The aggregation rules for both model parts are the same. The set of tokens removed (added) by firing one of the input (output) transitions must not be changed by the aggregation. The strong relationship between net elements in the work plan models and the structural model is kept after the aggregation.

Low-level Systems and Basic Skeleton

In the following step the aggregated subsystems are derived. They are called *low-level systems* and denoted by LS_i $(i \in \{1, \ldots, n\})$. Each LS_i is obtained from N by aggregating all subsystems in S except for S_i. The basic skeleton BS is obtained from N by aggregating all subsystems in S. The aggregation rules described in the previous section are applied during this step. Figure 5 shows the low-level system $LS_{assembly}$ of our manufacturing system example.

4 Iterative Throughput Approximation Algorithm

In this section the low-level systems and the basic skeleton are used to iteratively compute an approximation of the model throughput. The applied technique is based on the response time approximation method presented in [5, 6, 10].

The results of the iterative aproximation algorithm are independent from the initial service rates of the

[1] Please note that $IT_i \cap OT_i = \emptyset$ does not necessarily hold

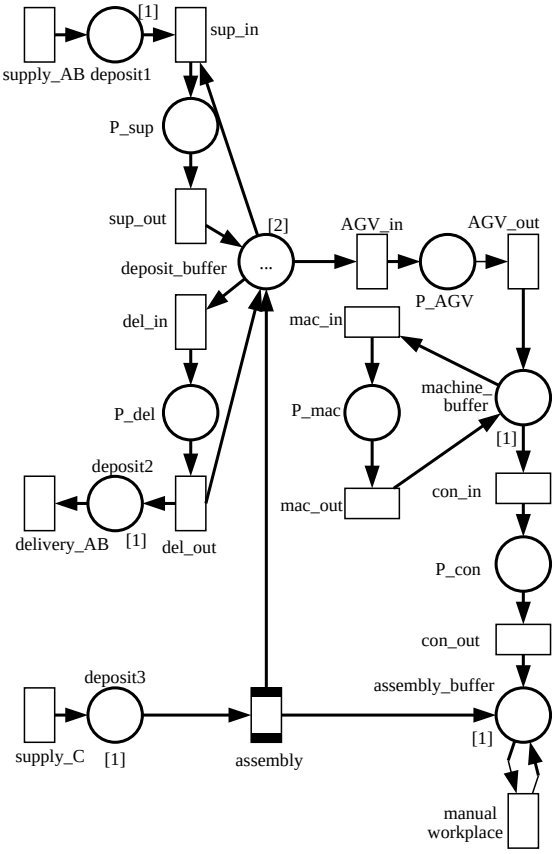

Figure 5: Low-level system example

input and output transitions of the aggregated model parts. In the non-aggregated parts the transitions keep the initial service rate of the original model. For each LS_i the throughputs of the output transitions $t \in OT_i$ for each workpiece (color) are computed. Then the service rates of the corresponding transitions of the basic skeleton are changed for each workpiece, such that the throughput of both transitions is the same. The service rates of the output transitions $t \in OT_i$ in all LS_j with $j \neq i$ is adjusted according to the values in the basic skeleton. The procedure is repeated until convergence is reached.

Algorithm
derive LS_i $(i \in \{1, \ldots, n\})$ and BS
initiate all $t \in (IT_i \cup OT_i)$ in LS_j $(j \neq i)$
and in BS with any service rate μ_t
keep the original service rates for all
$t \in (IT_i \cup OT_i)$ in LS_i
repeat
for $k := 1$ **to** n **do**
compute throughput of all $t \in OT_k$
by computing utilisation
repeat
change the service rates μ_t in BS
compute throughput of all $t \in OT_k$ in BS
until throughputs are equal
change the new service rates for the corresponding transitions t in all LS_i $i \neq k$
end for
until convergence

In our example (Figure 1) the throughput of the transition $delivery\_AB$ has to be computed. The underlying CTMC of the original model has 47600 states while the basic skeleton only has 12400 states. The initial service rates of all input and output transitions of the aggregated parts in the low-level systems and the basic skeleton are 1. Convergence has been reached after only four iterations. Table 1 shows the approximately computed throughputs of output transitions. The throughput value of the $delivery\_AB$ transition in the original model is 0.00975, computed with TimeNET$_{MS}$ [11]. The result of the approximation algorithm is 0.0093. Thus, the error of the approximation result is less then 5%.

5 Conclusion

This paper presents the extension of a response time approximation technique to colored stochastic Petri nets. The application focuses on the planning and design of manufacturing systems, where an estimation of the throughput is often sufficient and the systems are too large to be numerically analyzed. A dedicated class of Petri nets for the modeling of manufacturing systems is used. Their modular structure can be exploited in order to find a partitioning of the original model. The paper proposes to build one subsystem for each page of the hierarchical model. Moreover, the paper presents simple aggregation rules that are necessary to make the generated subsystems analyzeable. This technique makes use of the seperate structure and work plan models. Finally it is shown how to compute the throughput of an investigated system. An iterative approximation algorithm [5] is extended to colored Petri nets.

The described techniques have been applied to an example. The approximation error was acceptable. In the future we will investigate if more complicated aggregation rules can decrease the approximation error.

References

[1] G. Ciardo and K.S. Trivedi. A decomposition approach for stochastic petri net models. In *4th Int. Workshop on Petri Nets and Performance Models*, pages 74–83. IEEE-CS Press., 1991.

[2] P. Buchholz. Aggregation and reduction techniques for hierarchical gcspns. In *5th Int. Work-*

Subsystem S_k	Transition t	Workpiece	Throughput of $t \in OT_k$ in LS_k	μ_t in BS	Throughput of $t \in OT_k$ in BS	Throughput of *delivery_AB*
robot	sup_out	A	0.00460	0.1667	0.00466	
		B	0.00460	0.1667	0.00466	
	del_out	A	0.00741	0.1818	0.00727	
		B	0.00626	0.1818	0.00618	
AGV	agv_out	A	0.00460	0.1667	0.00466	
		B	0.00460	0.1667	0.00466	
machine	mac_out	A	0.00105	0.0434	0.00105	
		B	0.00132	0.0476	0.00134	
conveyor	con_out	A	0.00554	0.0178	0.00542	
		B	0.00556	0.0196	0.00561	
assembly	fit_out	A	0.00465	0.0416	0.00470	
		B	0.00438	0.0416	0.00444	0.0093

Table 1: Results after the final iteration

shop on Petri Nets and Performance Models, pages 216–225. IEEE-CS Press., 1993.

[3] Y. Li and M. Woodside. Iterative decomposition and aggregation of stochastic marked graph Petri nets. In G. Rozenberg, editor, *14th Int. Conference on Application and Theory of Petri Nets*, LNCS 674, pages 325–349. Springer Verlag, 1993.

[4] P. Ziegler and H. Szczerbicka. A structure based decomposition approach for GSPN. In *6th Int. Workshop on Petri Nets and Performance Models*, pages 261–270. IEEE-CS Press., 1995.

[5] C. J. Pérez-Jiménez and J. Campos. A response time approximation technique for stochastic general P/T systems. In *Computational Engeneering in Systems Applications (CESA '98)*, April 1998.

[6] J. Campos, J. M. Colom, H. Jungnitz, and M. Silva. Approximate throughput computation of stochastic marked graphs. *IEEE Transactions on Software Engineering*, 20(7):526–535, July 1994.

[7] A. Zimmermann, S. Bode, and G. Hommel. Performance and dependability evaluation of manufacturing systems using Petri nets. In *Workshop Manufacturing and Petri Nets at 17th Int. Conference on Application and Theory of Petri Nets*, 1996.

[8] K. Jensen. Coloured Petri nets. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Advances on Petri Nets*, LNCS 254, pages 248–299. Springer Verlag, 1986.

[9] M. Ajmone Marsan, G. Balbo, and G. Conte. A class of generalized stochastic Petri nets for the performance analysis of multi-processor systems. *ACM Transactions on Computer Systems*, 2(1), May 1994.

[10] C. J. Pérez-Jiménez, J. Campos, and M. Silva. Sate machine reduction for the approximate performance evaluation of manufacturing systems modelled with cooperating sequential processes. In *1996 IEEE International Conference on Robotics and Automation*, pages 1159–1165, April 1996.

[11] A. Zimmermann and J. Freiheit. TimeNET$_{MS}$ - an integrated modeling and performance evaluation tool for manufacturing systems. In *1998 IEEE Int. Conference on Systems, Man and Cybernetics*, October 1998. (to appear).