Subnetwork Time Distributions as a Means for Multilevel Evaluation of Generalized Stochastic Petri Nets

Guenter Klas , Reinhard Matuschka

Siemens Corporate R&D, ZFE IS SYS 13
Otto-Hahn Ring 6, 8000 Munich 83, Germany

Abstract

In this paper a new hierarchical evaluation procedure for Generalized Stochastic Petri Nets (GSPN) is being presented. It is based on extensions of Flow Equivalent Aggregation (FEA). At every level of hierarchy, subnets are approximated by substitute networks. As criterion for the similarity of the networks, the Subnetwork Time Distribution (STD) is used which is the sojourn time distribution of a token X in a subnet conditioned on the token distribution at the epoch of arrival of X and the context into which the net is embedded for determining the STD. The computation of a substitute network's type and parameters is outlined. The performance of this technique called FEAD (FEA based on Subnetwork Time Distribution) is discussed by means of examples.

1 Introduction

1.1 Background

Generalized Stochastic Petri Nets (GSPN) [14] are often used besides queueing networks in performance and performability evaluation of computers and communication networks. Since GSPNs are usually evaluated by solving the global balance equations, their numerical treatment soon gets unmanageable with an increasing state space. Two basic approaches exist to cope with the state space explosion.

First, the problem may be shifted to finding a powerful numerical solution method on matrix level. Then, usually the whole state space is constructed and the global balance equations are approximately solved [7,18,5,10,15]. The approximation quality often depends on the degree of nearly completely decomposability of the nets.

Second, a homogeneous hierarchical solution of GSPNs may be performed by replacing submodels at every level of hierarchy by less complex substitute nets. This avoids the construction of the whole state space. The basic aggregation step consists of finding a substitute net which behaves similar to the original submodel. This may be done by analyzing a submodel under certain conditions, e.g. open subnet with a predefined token input process [12] or a directly fed back, closed submodel with a fixed token population. The latter approach stems from Norton's theorem or Flow Equivalent Aggregation FEA [6,16] which is only exact for nets with local balance. FEA has been applied to non-product form nets on a heuristic basis [2,9,1,17]. It was extended by simulative preanalysis of subnets e.g. in [3].

In [11] FEA was considered for GSPN submodels with internal resources and blocking of output transitions. The authors of [11] propose a special aggregate network which takes into account (i) the delay which a token experiences until it is accepted by the submodel's input and (ii) the token's processing time in the submodel. Therefore, the aggregated net is similar to a Generalized Branching Erlang net [16] with 1 and 2 stages. These considerations naturally lead to the approach outlined in the following sections when the structure of the GSPN submodel is allowed to be quite general. The original submodel is replaced by a substitute network which shows a similar stochastic behaviour under the same stimulating conditions. A crucial point is, therefore, the definition of similarity.

1.2 Contribution

We propose a hierarchical extension of standard FEA called FEAD (FEA based on Subnetwork Time Distribution STD). FEAD is an analytical

TH0386-3/91/0000/0042/$01.00 © 1991 IEEE

hierarchical solution method for computing approximately the steady state solution of a large GSPN net. It differs from former approaches, e.g. [3], in four respects:

1) The *types of substitute networks* (or aggregates) are different.
2) As criterion for the similarity of original subnet and aggregate, a token's load dependent sojourn time in the net under *varying token distributions* at the epoch of the token's arrival is used.
3) The STD conditioned on the submodel's *GSPN neighbourhood* at every level of hierarchy is considered.
4) The aggregation is applied at *multiple levels of hierarchy*. The model hierarchy seems to be more general then the one recently proposed in [4].

With FEAD the submodels are replaced by substitute networks by using the subnetwork time distribution as matching criterion. The STD is computed in dependence on the submodel's context and its arrival point token distribution.

The rest of the paper is organized as follows: in chapter 2 some definitions are introduced. Then in chapter 3 the STD is defined and its computation discussed. The estimation of a substitute network's type and parameters is shown in chapter 4. The FEAD algorithm is summarized in chapter 5. After providing numerical results in chapter 6 we will give some conclusions in chapter 7.

2 Definitions

A GSPN model G is hierarchically decomposed into subnets. G is represented by a tree whose vertices are labelled by subnetwork names. The root of the tree, G, is of hierarchy level 0. Define for a network N S(N) as set of networks that are direct successors of N. Define for all $T \in S(N)$ $P(T)=\{N\}$, which means that N is direct predecessor of T. Since we use an "is part of" semantic for the edges of the tree, all nets $T \in S(N)$ are part of the net P(T), but $\cup_{T \in S(N)} T$ is not necessarily P(T). A subnet N has to adhere to the following restrictions:

1) Tokens enter subnet N via one entry place $\in N$ and leave net N via one exit transition.
2) N does neither produce nor consume tokens.
3) There is no direct feedback of N.

Definition 2.1 *Neighbourhood*
Let N be a subnet. The neighbourhood $v(N)$ of N is defined as

$$v(N) = P(N) \cup S(P(N)) \setminus \{N\}.$$

Definition 2.2 *Neighbourhood test network*
A network M is called a neighbourhood test network if it is closed and consists of the series connection of a subnet N and a load dependent server L with exponentially distributed service times.

Definition 2.3 *Isolation*
A net N is considered under isolation as net N^i if it is separated from the original net and if a direct feedback is applied from its exit transition to its entry place.

Definition 2.4 *R-, T-tokens, population*
Tokens which model local resources in a net are defined R-tokens. Tokens that enter and leave a net are denoted T-tokens. Since, with GSPN, tokens are indistinguishable, the distinction refers only to the population of a net which is counted in T-tokens.

We refer to T-tokens simply as tokens and mention R-tokens explicitly. The state space of a closed net N with p tokens is denoted Z_p or $Z_p(N)$. The restriction of a set of markings I to the places of a net N is denoted $[I]_N$. Places are annotated with there capacity c if $c \neq \infty$, e.g. *place1,c*. *#S* denotes the number of tokens in place *S*. In the sequel the STD is defined and its computation is outlined.

3 The Subnetwork Time Distribution

We are interested in the time it takes for a token X to proceed through a subnet N that holds k-1 tokens when X enters the subnet.

3.1 Definition of the Subnetwork Time Distribution STD

Definition 3.1 *Subnetwork Time Distribution*
Let N be a subnetwork of a given GSPN model. $D_{N,k}(t,s)$ is defined as the distribution of time that a token X, initially placed in the entry place of N, needs to proceed through the network N when k-1 tokens are in the subnet N at the time of the entrance of token X and when N is embedded in a GSPN context. The strategy s fixes the set $I_{F,k-1}$ of arrival instant states with k-1 tokens in N and their distribution F_{k-1}.

Definition 3.2 *F-net and STD-net*
An F-net serves for the computation of an arrival instant distribution F_{k-1} and has a state space $Z_{k-1} = I_{F,k-1}$. A STD-net is an augmentation of the F-net and is isomorph to an absorbing Markov chain. It serves for the computation of the STD. For its initial markings I_{STD} holds $[I_{STD}]_N = I_{F,k-1}^*$, the

markings $I_{F,k-1}$ with 1 additional token (the kth one) in the entry place of net N.

Since F is a priori unknown (although in principle computable from the original net) we use heuristics for establishing a distribution F. For a fixed value of k we consider the following strategies s:

WA Worst Case Analysis

Set $I_{F,k-1}$ is fixed by a marking with k-1 tokens initially located in the entry place of N. The support of F_{k-1} is a singleton.

SY Stationary Distribution Analysis

The F-net consists of net N^i with k-1 tokens circulating in it. This yields F_{k-1} of the markings of a state space Z_{k-1}. Before computing $D_{N,k}(t,SY)$, k-1 tokens initially are located in the places of N according to F_{k-1}.

ND Neighbourhood Dependence Analysis

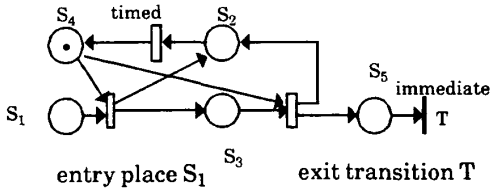
The F-net consists of the neighbourhood test network M which is constructed of subnet N and a load dependent exponentially distributed server L that has been gained by means of FEA of the neighbourhood v(N). The following preliminary analysis is performed in order to fix the initial distribution F_{k-1} of the $i=k-1$ tokens in N:

Given p tokens in M, there will be, on an average, $n(p) \in \mathbb{R}$ tokens in N and p-n(p) tokens in L after a stationary analysis of M. The initial marking seen by the arriving kth token can only comprise **exactly** $i \in \mathbb{N}$ tokens in N. Assuming $n(p)<i$ and $n(p+1)>i$, this can be achieved by a random initial marking that takes values from the state spaces Z_p and Z_{p+1} of net M with probabilities g(p) and g(p+1) respectively, such that the mean number of tokens in N in the initial marking (seen by the arriving token $k=i+1$) equals $i \in \mathbb{N}$. A random initial marking of M takes a state j from Z_p with probability g(p)*P{state j occurs in Z_p} and from Z_{p+1} with probability g(p+1)*P{state j occurs in Z_{p+1}}. State space Z_p is chosen with probability $g(p) = \{n(p+1)-i\}/\{n(p+1)-n(p)\}$ and state space Z_{p+1} with probability $g(p+1):=1-g(p)$, because $i=g(p)n(p) + g(p+1)n(p+1)$. In the following section the numerical computation of a STD is considered.

3.2 Computation of the Subnetwork Time Distribution $D_{N,k}(t,s)$

$D_{N,k}(t,s)$ is computed from an absorbing continuous time Markov chain (CTMC). For its initial states I_{STD} holds: $[I_{STD}]_N = I_{F,k-1}{}^+$. The absorbing states are defined by the fact that token X, initially located in the entry place of net N, has just left N. Since a close tracking of token X, which is indistinguishable from other tokens in GSPN net N, is quite complex, we consider the following approximation for the computation of $D_{N,k}(t,s)$: an absorbing state is reached after the kth firing of the exit transition of N.

$D_{N,k}(t,s)$ is computed at equidistant time points $t_i = i*\Delta t$ for $i = 0...c$ as time to absorption of the absorbing CTMC by means of the randomization technique [8] and, in the case of an acyclic Markov chain, by means of the algorithm ACE [13] which yields the distribution also in a symbolic representation. Now we show the constructions needed to compute $D_{N,k}(t,s)$ for the strategies WA, SY and ND with $k=2$ for an example. Figure 3.1a shows the submodel N considered.



All exponential transitions have rate 1.

Figure 3.1a: Network N

Strategy WA:

The initial marking seen by the arriving second token is determined by 1 token in place S1.

Figure 3.1b shows the STD-net necessary to compute $D_{N,2}(t,WA)$ as the distribution of the time by which place Count is empty.

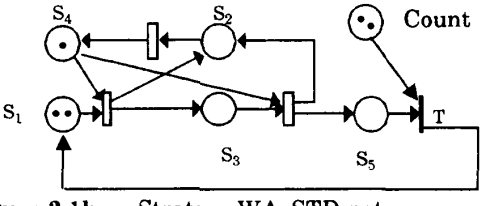


Figure 3.1b: Strategy WA, STD-net

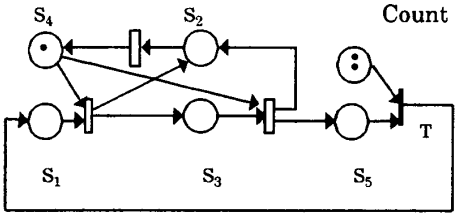
Strategy SY:

Since the stationary distribution of k-1 tokens in the closed submodel is used as the arrival point distribution which is seen by the arriving kth token, stationary analysis for $k-1=1$ circulating token of N^i supplies the stationary distribution F_1 of the states $I_{F,1}$ seen by the second token:

$P(a_1,a_2,a_3,a_4,) := P\{a_i \text{ tokens in } S_i, i=1...4\}$

$P(0,1,1,0) = 0.25 \quad P(1,1,0,0) = 0.25$
$P(0,0,1,1) = 0.25 \quad P(1,0,0,1) = 0.25 \qquad (3.1)$

Dimension of state space Z_1: 4. Figure 3.1c shows the STD-net. For each state $j \in I_{F,1}$ an immediate transition t_j has to be produced with probability $P(a_{1,j},a_{2,j},a_{3,j},a_{4,j})$ for the state $(a_{1,j},a_{2,j},a_{3,j},a_{4,j})$, whose multiple arcs with weights $a_{i,j}$ lead to places S_i. Moreover, a place Start and arcs from Start to all transitions t_j are constructed. $D_{N,2}(t,SY)$ is then computed as the time to absorption of a MC with initial states (3.1), but with 1 additional token in place S1.



Below: example for state (0,1,1,0)

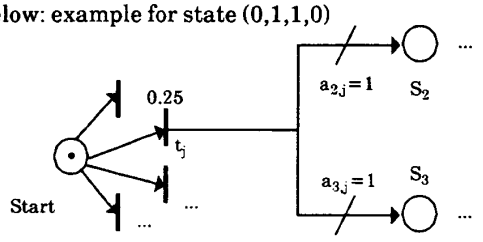


Figure 3.1c: Strategy SY, STD-net

Strategy ND:

The neighbourhood of net N is aggregated to a load dependent server (figure 3.1d). The initial distribution F_1 of 1 token in net N is determined by stationary analysis for one and two tokens of the neighbourhood test network M which supplies the stationary distribution of states of the state spaces Z_1 and Z_2.

1 token : mean number of tokens in N: 0.78
2 tokens: mean number of tokens in N: 1.69.

$I_{F,1} = Z_1 \cup Z_2$.

Next, we proceed analogously to Strategy SY, but for both state spaces Z_m, $m = 1,2$. For each state j_m an immediate transition t_{jm} is appended with probability $g(m)P\{j_m\}$. The multiple arcs of these transitions with weights $a_{i,jm}$ lead to places S_i. A place Start and arcs from Start to all transitions t_{jm} have to be produced. This constitutes the STD-net. $D_{N,2}(t,ND)$ is computed analogously to $D_{N,2}(t,SY)$ using the initial states from spaces Z_1 and Z_2 augmented by one token in place S1.

So far, a GSPN subnet N is characterized by means of its Subnetwork Time Distribution. Using this STD as matching criterion we compute an optimum substitute network which is less complex than the original subnet.

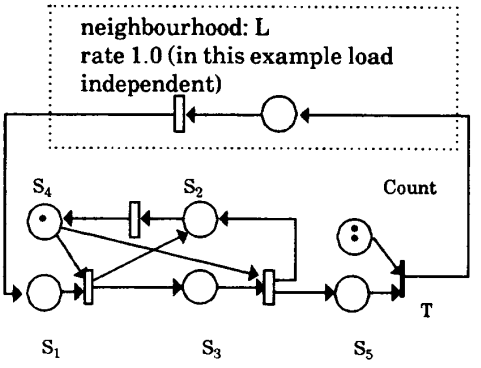


Figure 3.1d: Strategy ND, part of the STD-net

4 Selection of the substitute network

Problem statement:

Given subnet N and STD $D_{N,k}(t_i,s)$, $k=1...K=capacity(N)$, $t_i=i*\Delta t$, $i=0...c$ under a strategy s, we have to find a substitute network SN that shows similar stochastic behaviour and has a smaller state space than N.

Since we use STDs as matching criterion, substitute network types should feature small model complexity and practical computability of these distributions. The subnetwork time distribution $D_{Ei,k}(\mathbf{x};t,s)$ for a substitute network type E_i is given symbolically or numerically in dependence on the parameters $\mathbf{x} = [x_1, ..., x_{maxi}]$ with maxi as number of parameters of the network type E_i.

The subnetwork time distribution of both the original net N and the aggregate net E, $D_{N,k}$ and $D_{E,k}$ respectively, vary over the range of number of tokens in the net. Starting from the STDs $D_{N,d} = D_{N,d}(j*\Delta t)$, $j=0...c$ of a token when arriving as the dth customer at net N, we determine an optimum aggregate network type E_0 from a set of types $\{E_i\}$, $i=1...m$. The optimum type E_o is determined only from the equivalent service time distribution of submodel N, i.e. $D_{N,1}(j*\Delta t)$, $j \geq 0$. After this type is fixed, the aggregate net's parameters are adjusted to account for load dependence, such that $D_{N,d}(j*\Delta t) = D_{Eo,d}(j*\Delta t)$, $d=2...K$.

The type of aggregate network E_o is determined by sequencing through the set $\{E_i\}$, recording new minimum types and stopping even before the end of the set is reached if the Euklidian norm of the two vectors $\mathbf{D}_{N,1}$ and $\mathbf{D}_{Ei,1}$ is less than a prescribed minimum. We start the ordered set $\{E_i\}$ with E_1 as a load dependent exponential server. This yields the equivalent mean service time of submodel N. This parameter is used for types E_i, $i \geq 2$ to get a close initial guess $\mathbf{x}^i$ in algorithm 4.1. We use the following sequence:
$\{E_i\} = \{Er_1, Er_2, Er_3, Er_4, BE_2, BE_3, H_2, H_3, C_2\}$ with Er_k denoting an Erlang-k server, BE_k denoting a Branching Erlang-k server, H_k denoting a hyperexponential server with k branches and C_2 a Cox-2 server.

Algorithm 4.1:

begin
for all types $E_i \in \{E_i\}$
minimize $\|\mathbf{D}_{N,1} - \mathbf{D}_{Ei,1}\|$
$\rightarrow$ optimum type E_o.
for d = 2 to K
determine parameters of E_o for load d.
end

The optimal parameters of the aggregate net E_i for load d: $x_1, x_2, \ldots, x_{maxi}$ are computed by minimizing the following functional:

(4.1) $$min_{(x_1,\ldots,x_{maxi})} f(x_1,\ldots,x_{maxi}) =$$

$$= \sum_{j=0}^{c-1} [D_{N,d}(j^*\Delta t) - D_{E_i,d}(x_1, \ldots, x_{maxi}, j^*\Delta t)]^2 .$$

This constitutes a *nonlinear* optimization problem. Let F_d^k denote the Jacobian matrix for net type E_k and a load of d tokens with

$$F_d^k(i,j) = \frac{\partial D_{E_k,d}(x_1, x_2, \ldots, x_{maxk}, i^*\Delta t)}{\partial x_j}$$

and i = 0, ..., c-1, j = 1, ... maxk.
The flow time distribution $\mathbf{D}_{Ei,d}(\mathbf{x})$ is linearized as

$$\mathbf{D}_{E_i,d}(\mathbf{x}+\Delta\mathbf{x}) = \mathbf{D}_{E_i,d}(\mathbf{x}) + F_d^i(\mathbf{x})\Delta\mathbf{x} + O(\Delta\mathbf{x}^2)$$

The solution of (4.1) is then iteratively possible by the following algorithm:

Algorithm 4.2
$\mathbf{x} = \mathbf{x}^i$ /* initial parameter guess */
do
solve the following *linear* optimization problem (4.2) for $\Delta\mathbf{x}$:

$$min\, f(\Delta\mathbf{x}) =$$

$$= \sum_{j=0}^{c-1} [D_{N,d}(j^*\Delta t) - D_{E_i,d}(\mathbf{x}, j^*\Delta t) - F_d^i(j,.)\Delta\mathbf{x}]^2$$

$\mathbf{x} = \mathbf{x} + \Delta\mathbf{x}$
while($\|\Delta\mathbf{x}\| > \varepsilon$ AND $f > \eta$). For problem (4.2) the normal equations are established and solved for $\Delta\mathbf{x}$

$$F_d^{i\,T} F_d^i \Delta\mathbf{x} = F_d^{i\,T}(\mathbf{D}_{N,d} - \mathbf{D}_{E_i,d}(\mathbf{x})).$$

In the next section, the complete algorithm is discussed.

5 Summary of the algorithm

Two variants of FEAD are presented. Let N_h be the set of networks of hierarchy level h.

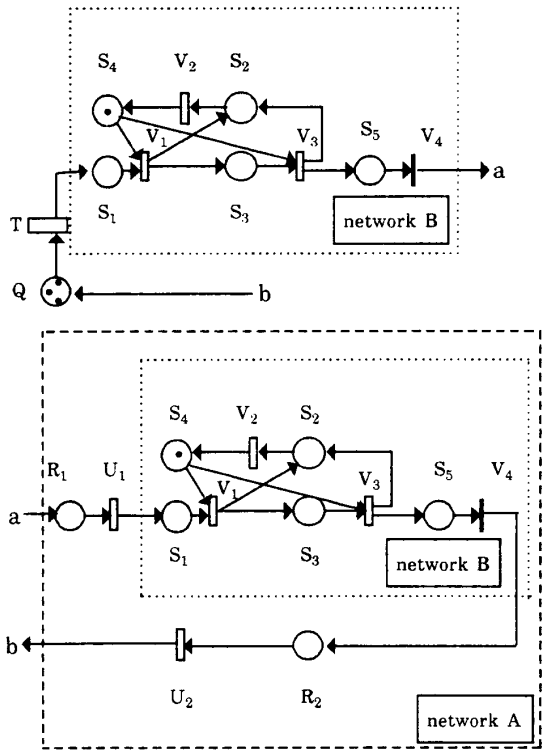
Algorithm 5.1: Variant One
FEAD *without* dependence on neighbourhood of subnetworks, Strategy s = WA or s = SY.
begin
for hierarchy level h = H = max(h) **to** 1 **do**
for all $N \in N_h$ **do**
for k = 1 **to** capacity(N) **do**
begin
Calculate $D_{N,k}(t,s)$ from the STD-net,
Choose a substitute network SN for N according to algorithm 4.1.
end
Perform stationary analysis for the aggregated main network.
end

Algorithm 5.1: Variant Two
FEAD *with* dependence on neighbourhood of subnetworks, Strategy s = ND.
begin
for hierarchy level h = H **to** 1 **do**
for all $N \in N_h$ **do**
begin
if $|S(P(N))| > 1$ **then**
Apply FEA [16] to all $T \in S(P(N)) \setminus \{N\}$
Apply FEA to the now aggregated $P(N)\setminus\{N\} \Rightarrow$ load dependent server L
Establish the neighbourhood test network M.
for k = 1 **to** capacity(N) **do**
begin
Calculate $D_{N,k}(t,ND)$ from the STD-net.
Choose a substitute network SN for N according to algorithm 4.1.
end
end
Perform stationary analysis for the aggregated main network.
end

6 Numerical results

Two examples that are kept simple for the sake of clarity are analyzed. Although they do not justify any aggregation due to their small complexity, they provide some insight into the algorithm's performance.

Example 1:
Consider the GSPN model in figure 6.1.



Trans. U_1 has rate 4; rate r of T varies;
other transitions have rate 1, cap. of places $= \infty$

Figure 6.1: Example of a GSPN model which is hierarchically solved by FEAD

The following algorithms are compared:

1) exact solution (E)
2) Flow Equivalent Aggregation (F)
3) FEAD Variant One WA (WA)
4) FEAD Variant One SY (SY)
5) FEAD Variant Two ND (ND).

As performance criterion we regard the deviation

$e := result_{approximate\ solution} - result_{exact\ solution}$

in

a) mean number of tokens in place Q,
b) mean throughput of transition T,
c) probability of finding i tokens in place Q prob(#Q=i) for i = 0 ... 3.

Moreover, the rate r of transition T which influences the coupling degree of the submodels is varied. The results are presented in figures 6.2 and 6.3. In this example, we find that Flow Equivalent Aggregation shows a bad performance, especially in the mean number of tokens in place Q and in prob(#Q=i), in comparison to the other aggregation algorithms. This indicates that the subnetwork time distribution is a good criterion for the characterization of subnets at several levels of hierarchy. Furthermore, the different strategies with FEAD may be discussed. We find that FEAD-SY behaves similarly to FEAD-ND, although ND is slightly better than SY. Surprisingly FEAD-WA shows the best performance in computing the throughput of T and prob(#Q=0) and is nearly as good as ND and SY in the other values. This indicates that the positive effect gained by using the STD as criterion for network similarity is much greater than the supplementary advantage stemming from a consideration of the network's neighbourhood.

Next, the aggregation results are considered with respect to the coupling degree of the models which is heavily influenced by the firing rate of transition T. Figures 6.2 and 6.3 show that standard FEA performs worst with respect to the other algorithms when the ratio of the submodel input rate r to the submodels' internal rates (=1.0) is in the interval [0.1,...,1]. This effect is alleviated when FEAD with various strategies is used.

From a computational point of view FEAD-WA is the most efficient FEAD variant.

Example 2:

Figure 6.4 shows the simple model of a multiprocessor system with 3 processors. The model is decomposed into processor and cache system and the bus and memory subsystem SNA. Model SNA, in turn, holds a submodel SNB which accounts for the memory system. Transitions and the model's parameters are defined in table 6.1. The application of algorithm 5.1 Variant One with strategy WA yields the following results for the throughput of transition T1:

exact result:	0.1068	error in %
FEAD-WA:	0.1024	4.1
FEA:	0.1124	5.2

The subnets are approximated by the following substitute nets where λ(i) denotes the rate parameter of an exponential phase for a load of i tokens:

N:	SN:	λ(i): i=1	2	3
SNB	Erlang-2	0.10	0.104	0.112
SNA	Erlang-4	0.13	0.156	0.174

Figure 6.5 compares the subnetwork time distributions $D_{.,d}(t)$ of the original subnet SNA and the substitute nets for token populations $d \in \{1,2,3\}$.

7 Conclusions

We have presented an approach for the hierarchical solution of Generalized Stochastic Petri Nets using a step by step aggregation of GSPN submodels. Subnetworks are approximated by substitute networks using the Subnetwork Time Distribution STD introduced in chapter 3 as criterion for the stochastic equivalence of subnet and aggregate. An STD equals the sojourn time distribution of a token in a subnet dependent on the token distribution F seen by the arriving token and dependent on the subnet's GSPN neighbourhood.

A problem consists in the computation or efficient estimation of the a priori unknown arrival point token distribution F. We approached this by using heuristic strategies. In chapter 4 we discussed the determination of the optimum substitute network type and the estimation of the load dependent parameters. In our prototype implementation we computed the optimum parameters of a series of substitute network types by nonlinear optimization. For nontrivial substitute nets we compute e.g. the Jacobian of an STD by numerical differentiation and the STD by numerical convolution. Although this approach of sequencing through a series of net types is readily extended to new substitute network types, which is suitable in an experimental phase, it seems to be more appropriate to restrict the aggregate network types to a single class, e.g. 2 stage Coxian networks for which the parameter estimation problem is directly solved either numerically or analytically.

The algorithm presented in chapter 5 was implemented in a tool. Since major parts of this tool like compiler, reachability generator, steady state solver and algorithms for the transient analysis were taken from a general GSPN analyzer developed at the authors`institute, modules with well defined interfaces were available. For this reason and for the sake of easy testing, most constructions necessary in algorithm 5.1 as e.g. applying a direct feedback to a subnet were performed on GSPN level by manipulating GSPN specification files. For a more efficient implementation a direct manipulation of the generator matrices is recommended.

The additional improvement introduced by taking into account a subnet's neighbourhood was shown to be relatively small with respect to the dominating improvement achieved by accounting for the second moment in a subnet's equivalent serive time distribution. In the considered example, standard Flow Equivalent Aggregation yields worse results than the aggregation algorithms presented in this paper which motivates further research in this area.

Acknowledgement

Thanks are due to the referees and Helen Hoffmann for reviewing the final paper, to Roland Lepold from Siemens Labs Munich and to Prof. Eike Jessen from Technical University Munich.

References

[1] G. Balbo, S.C. Bruell and S. Ghanta, "Combining queueing networks and generalized stochastic Petri nets for the solution of complex models of system behaviour", IEEE Trans. on Computers, vol. 37, no. 10, pp. 1251-1268, 1988

[2] H. Beilner, "Structured Modelling - Heterogeneous Modelling", European Simulation Multiconference, Rome, June 1989.

[3] H. Beilner, P. Buchholz and B. Müller-Clostermann, "Experimente mit Ersatzdarstellungen unter Berücksichtigung der Verweilzeitverteilung", 4. GI/NTG Fachtagung, 29.9-1.10.87, Erlangen, "Messung, Modellierung u. Bewertung von Rechensystemen"

[4] P. Buchholz, "Numerical Solution Methods Based on Structured Descriptions of Markovian Models", Fifth Intern. Conference on Modelling Techniques and Tools for Computer Performance Evaluation, Torino, Italy, Feb. 13-15, 1991, pp.242-258

[5] W.L. Cao and W.J. Stewart, "Iterative Aggregation / Disaggregation Techniques for Nearly Uncoupled Markov Chains", Journal of the ACM, 32(3), July 1985, pp. 702-719

[6] K. M. Chandy, U. Herzog and L. Woo, "Parametric Analysis of Queueing Networks", IBM J. Res. Develop., Jan. 1975, pp.36-42

[7] P.J. Courtois and P. Semal, "Bounds for the Positive Eigenvectors of Nonnegative Matrices and for their Approximations by Decomposition", Journal of the ACM, 31(4), 1984, pp.804-825

[8] D. Gross, D. Miller, "The Randomization Technique as a Modeling Tool and Solution Procedure for Transient Markov Processes", Operations Research Vol.32 No.2 pp. 343-361 1984

[9] B.R. Haverkort and I.G. Niemegeers, "Using Dynamic Queueing Networks for Performability Modelling", Proceedings of the European Simulation Multiconference, June 10-13, 1990 Nuremberg, Germany

[10]M. Haviv, "Aggregation / Disaggregation Methods for Computing the Stationary Distribution of a Markov Chain", SIAM Journal of Numerical Analysis, vol. 24, no. 4, Aug. 87, pp.952-966

[11]H. Jungnitz and A. Desrochers, "State Space Reduction for Discrete Event Dynamic Systems", IMACS Symp. MCTS, Casablanca 1991

[12]G. Klas and U. Seidel, "A Subsystem Identification Algorithm for the Approximate Solution of Large GSPN Models", Proceedings of the Fifth Intern. Conference on Modelling Techniques and Tools for Computer Performance Evaluation, Torino, Italy, Feb. 13-15 1991, pp. 259-275

[13]R. Marie, A. Reibman, K. Trivedi, "Transient Analysis of Acyclic Markov Chains", Performance Evaluation 7(1987) pp.175-194

[14]M. Ajmone Marsan, G. Balbo, G. Conte, "A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems", ACM Transaction on Computer Systems, 2(2) 1984, pp.93-122

[15]B. Philippe, Y. Saad, W.J. Stewart, "Numerical Methods in Markov Chain Modeling", IRISA, Rennes, France, Report no. 495, Sept. 1989

[16]C.H. Sauer and K.M. Chandy, "Computer System Performance Modeling", Prentice Hall, Englewood Cliffs, 1981

[17]H. Szczerbicka, "A Combined Queueing Networks and Stochastic Petri Nets Approach for Evaluating the Performability of Fault-Tolerant Computer Systems", 1st Int. Workshop on Performability Modelling of Comp. and Computing Systems, Feb. 7-8, 1991, Univ. Twente, The Netherlands

[18]H.T. Vantilborgh, "Aggregation with an Error of $O(\varepsilon^2)$", Journal of the ACM, 32(1), 1985, pp.162-190

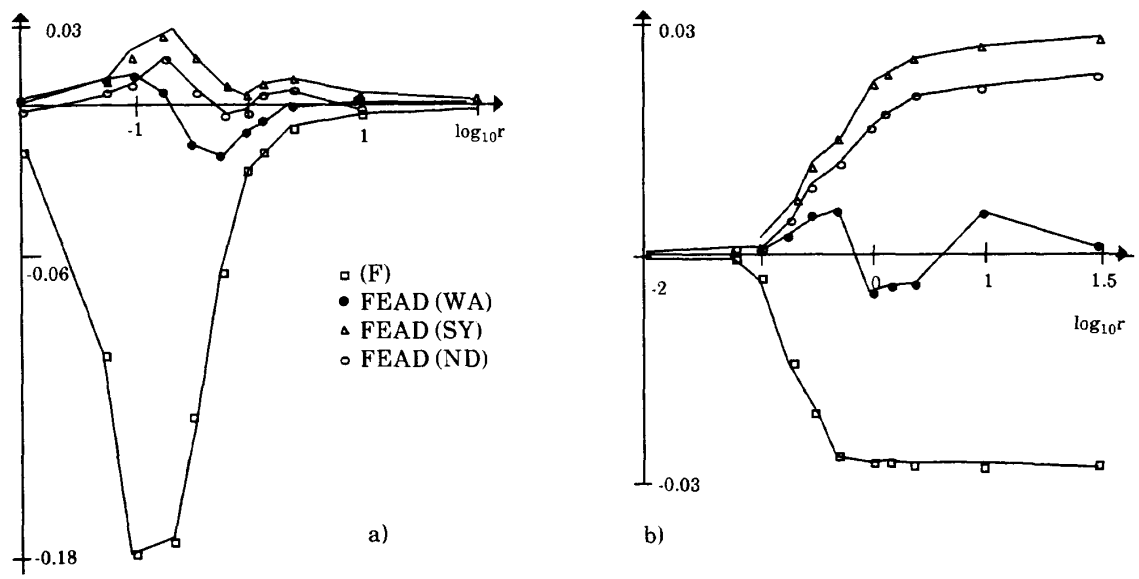


Figure 6.2: a) Absolute error of mean number of tokens in place Q, b) of mean throughput of T

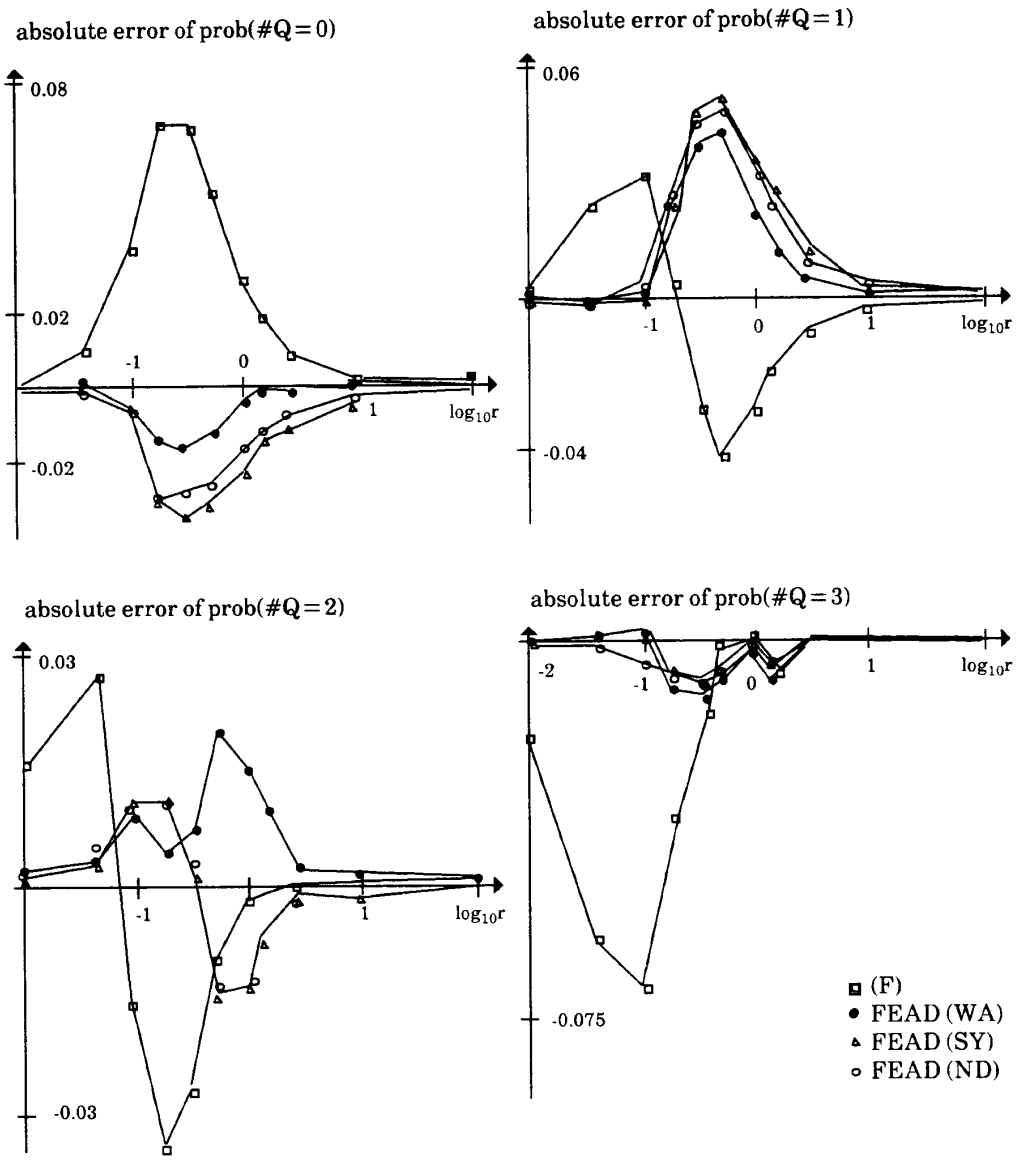
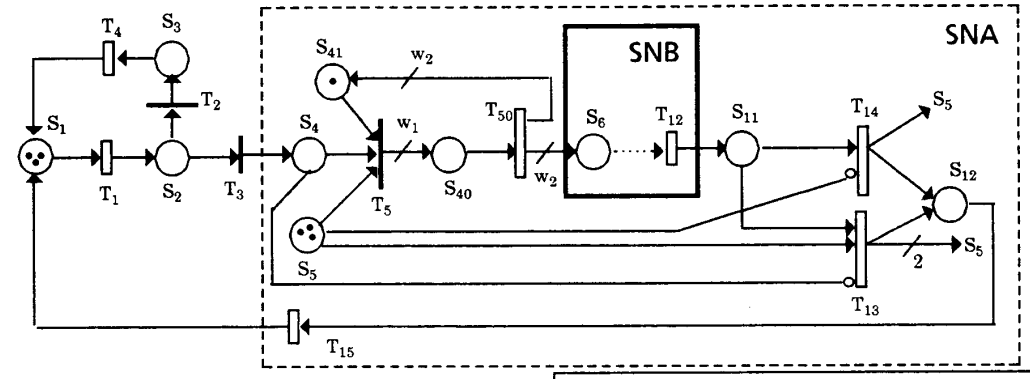


Figure 6.3: Absolute error of the aggregation algorithms in the probability of a fixed number of tokens in place Q (see the model in figure 6.1).



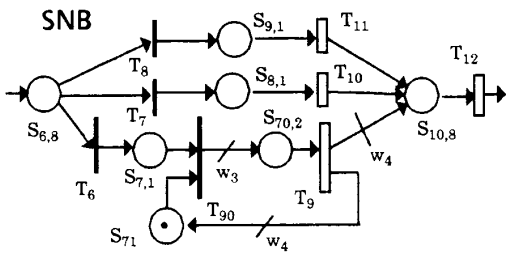


Figure 6.4: Example 2: multiprocessor

Table 6.1: Parameters of example 2

trans.	rate	probability	signification
T1	0.1*#S_1		rate for access to cache
T2		0.7	cache hit rate
T3		0.3	cache miss rate
T4	0.25		delay time on cache bus
T5		1.0	access to memory bus, 3 slots
T50	0.66		delay time of a request to mem.
T6	if(#S7 = 0) then 1.0 else 0.0		main memory access
T7	if(#S7 = 1) then 0.2 else 0.0		disk I/O 1
T8	if(#S7 = 1) then 0.8 else 0.0		disk I/O 2
T90		1.0	
T9	0.1		access time in main memory
T10	0.01		access time disk 1
T11	0.005		access time disk 2
T12	10.0		
T13	0.25		data reply on memory bus
T14	0.25		data reply on memory bus
T13, T14 model the scheduling of data replies w.r.t. requests			
T15	0.25		data reply on cache bus

Table 6.1: (cont.)

arc	weight
w1	2
w2	if(#S40 = 1) then 1 else 0
w3	2
w4	if(#S70 = 1) then 1 else 0
resources	
S_1	number of active processors
S_5	number of slots in the packet switched memory bus

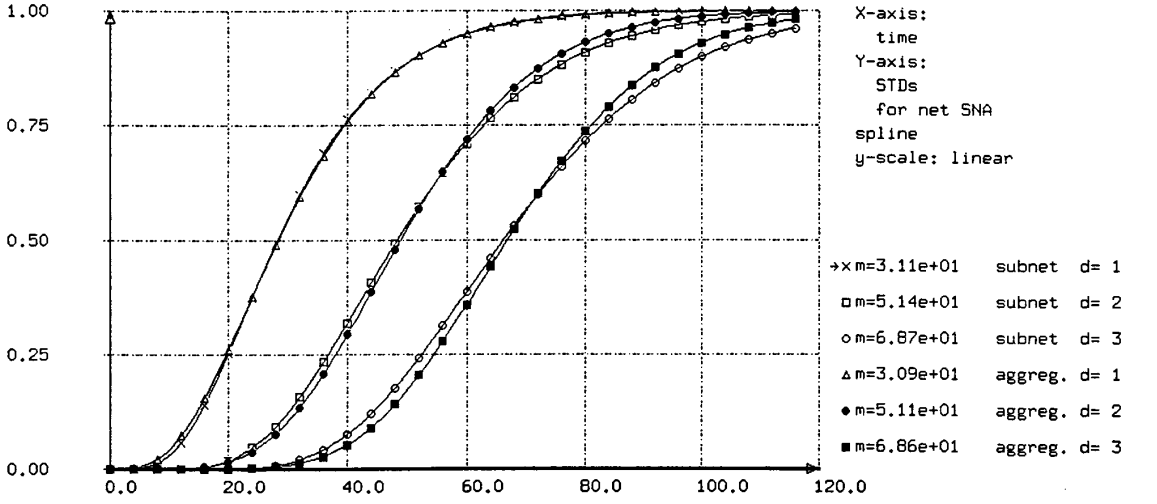


Figure 6.5: STDs for subnet SNA and aggregate: d = number of tokens for FEAD-WA, m = mean of the subnetwork time distribution