Performance Modeling using DSPNexpress*

Christoph Lindemann

Technische Universität Berlin
Institut für Technische Informatik
Franklinstr. 28/29
10587 Berlin, Germany

Abstract

This paper recalls the numerical solution method for Deterministic and Stochastic Petri Nets (DSPNs) and describes the graphical, interactive analysis tool DSPNexpress which has been developed at the Technische Universität Berlin since 1991. The software package DSPNexpress allows the employment of complex DSPNs for evaluating the performance and dependability of discrete-event dynamic systems such as computer systems, communication networks, or automated manufacturing systems. A DSPN of a memory consistency model for a multicomputer system with virtually shared memory is presented to illustrate the modeling power of DSPNs.

1 Introduction

Petri Nets in which transition firings can be augmented with a constant delay constitute an important tool for modeling various kinds of discrete-event dynamical systems, because such systems typically include activities with constant duration. Examples of activities which might have a constant duration are transfer times of data packets of fixed size in distributed computing systems, timeouts in real-time systems, and repair times of components in fault-tolerant systems. In Timed Petri Nets (TPNs) [20] all transitions have associated either constant delays (D-timed Petri Nets) or exponentially distributed delays (M-timed Petri Nets). Generalized Timed Petri Nets (GTPNs) have been introduced in [12] as an extension to TPNs. GTPNs contain immediate transitions firing without a delay and timed transitions firing after a deterministic delay. TPNs and GTPNs employ a discrete time scale for the underlying stochastic process. In TPN and GTPN timed transitions fire in three phases and the next transition to fire is chosen according to some probability distribution.

Deterministic and Stochastic Petri Nets (DSPNs) have been introduced in [2] as a continuous-time modeling tool which include both exponentially distributed and constant timing. In DSPNs, transition firing is atomic and the transition with the smallest firing delay is the next transition to fire. Under the restriction that in any marking at most one deterministic transition is enabled, a numerical solution method for calculating the steady state solution of DSPNs has been introduced. This method is based on the technique of the embedded Markov chain and requires the numerical calculation of transient quantities of continuous-time Markov chains defined by exponential transitions concurrently or competitively enabled with a deterministic

* This work was supported by the Federal Ministry for Research and Technology of Germany (BMFT) under grant ITR9003.

transition. Efficient computational formulas for the transient analysis of these Markov chains have been presented in [14]. DSPNs have been applied for modeling the performance of an Ethernet Bus LAN [3], and several fiber optic LAN architectures [5] as well as for deriving self-stability measures of a fault-tolerant clock synchronization system [18]. More recently, DSPNs have been employed for evaluating the performance of memory consistency models for multicomputer systems with virtually shared memory [17] and for an integrated performance/dependability analysis of an automated manufacturing system [15].

This paper recalls the numerical solution method for DSPNs and describes the graphical, interactive analysis tool DSPNexpress which which has recently become available [13]. The development of DSPNexpress has been motivated by the lack of a powerful software package for the numerical solution of DSPNs and the complexity requirements imposed by evaluating the performance and dependability of computer systems, communication networks, and automated manufacturing systems. The development of DSPNexpress has been motivated by the lack of a powerful software package for the numerical solution of DSPNs and the complexity requirements imposed by evaluating the performance and dependability of multicomputer systems, communication networks, and automated manufacturing systems. The remainder of this paper is organized as follows. Section 2 recalls the main steps of the numerical solution method for DSPNs. In Section 3 the software package DSPNexpress is described. A DSPN of a sequential memory consistency model for a multicomputer system with virtually shared memory is presented in Section 4. Finally, current research topics are outlined.

2 The Numerical Solution Algorithm for Deterministic and Stochastic Petri Nets

The numerical solution technique for computing steady-state marking probabilities of DSPN models introduced by Ajmone Marsan and Chiola [2] is based on the restriction that the DSPN does not contain markings in which two or more deterministic transitions are concurrently enabled. With this restriction they showed how to analyze the stochastic behavior of a DSPN using the technique of the *embedded Markov chain*. Sampling the stochastic behavior of the DSPN only at appropriately selected instants of time define regeneration points of a Markov regenerative stochastic process [9] in which the Markovian property holds. In case only exponential transitions are enabled the stochastic behavior of the DSPN is sampled at the instant of firing of an exponential transition. In case a deterministic transition is exclusively enabled the stochastic behavior of the DSPN is sampled at the instant of its firing. If a deterministic transition is competitively enabled with some exponential transitions, the stochastic behavior is sampled when either the deterministic or the exponential transition fires. If a deterministic transition is concurrently enabled with some exponential transitions, the stochastic behavior is sampled at the instant of time of firing the deterministic transition.

As a consequence, a change of marking in the DSPN due to firing of an exponential transition which is concurrently enabled with a deterministic transition is not represented in the embedded Markov chain by a corresponding state change. Thus, the embedded Markov chain typically contains fewer states than the number of tangible markings of the DSPN. The state space cardinalities of the DSPN and its embedded Markov chain are denoted by N and N', respectively. For example, the DSPN of the modified $E_r/D/1/K$ queue discussed below has $N = rK-1$ tangible markings, but its embedded Markov chain has only $N' = r(K-1)+1$ states. Assuming a DSPN contains markings in which a deterministic transition is competitively or concurrently enabled with some exponential transitions, the transition probabilities of its embedded Markov chain have to be derived by calculating transient quantities of the Markov chain defined by the firing rates of exponential transitions competitively or concurrently enabled with a deterministic transition and the firing probabilities of immediate transitions which are enabled directly after a firing of one of the exponential transitions. This continuous-time Markov chain has been referred to as the *subordinated Markov chain* (SMC) of a deterministic transition [14].

In the first step of the DSPN solution process the extended reachability graph consisting of tangible markings and directed arcs labelled with firing rates and weights causing the corresponding change of marking is generated. An efficient algorithm for generating the extended reachability graph for Generalized Stochastic Petri Nets has been introduced by Balbo, Chiola, Franceschinis and Molinar Roet [6].

The second step of the DSPN solution algorithm is the calculation of the transition probability matrix **P** of the embedded Markov chain and the conversion matrix **C**. For markings which enable a deterministic transition T_k competitively or concurrently with some exponential transition the corresponding entries of these matrices are derived by calculating time-dependent quantities of the SMC. Transient state probabilities of the SMC of the deterministic transition T_k determine the corresponding transition probabilities of the embedded Markov chain. According to [2] the transition probability $P(S_i \rightarrow S_j)$ between two states S_i and S_j of the embedded Markov chain is given by:

$$P(S_i \rightarrow S_j) = u_i \cdot e^{\mathbf{Q}\tau_k} \cdot \Delta_k \cdot u_j^T \tag{1}$$

In formula (1) **Q** denotes the generator matrix of the Markov chain subordinated to the deterministic transition T_k with firing delay τ_k. The term u_i denotes the i-th row unity vector of dimension N_k $(u_i \in \mathrm{R}^{1 \times N_k})$ where N_k is equal to the dimension of the generator matrix **Q**. The term u_j^T denotes the j-th column unity vector of dimension N' $(u_j^T \in \mathrm{R}^{N' \times 1})$. The matrix Δ_k denotes a transition probability matrix representing the feasible marking changes caused by a path of immediate transition firings which are enabled immediately after a firing of this deterministic transition. Nonzero entries of Δ_k are either "1" or given by the weights associated with conflicting immediate transitions. The matrix Δ_k is a rectangular matrix of dimension $N_k \, x \, N'$.

In case the deterministic transition T_k is concurrently enabled with some exponential transitions, the sojourn times in tangible markings of the continuous-time stochastic behavior of the DSPN caused by firings of exponential transitions during the enabling interval of the deterministic transition are not taken into account in the discrete-time embedded Markov chain. Thus, *conversion factors* are employed to derive the steady-state probability vector of the DSPN of dimension N from the steady-state probability vector of the embedded Markov chain of dimension N'. These conversion factors are calculated as the average sojourn times in a state S_j during the enabling interval of the deterministic transition T_k assuming this transition has become enabled in state S_i. These conversion factors are given by:

$$C(i,j) = \int_0^{\tau_k} u_i \cdot e^{\mathbf{Q}t} \cdot u_j^T \, dt \tag{2}$$

As shown in [14] the time-dependent quantities of formula (1) and (2) can be efficiently computed using the randomization technique [11].

Subsequently, in the third step of the DSPN solution process the steady-state solution, $\pi(.)$ of the embedded Markov chain is calculated by solving the linear system of its global balance equations.

$$\begin{aligned} &\pi \cdot \mathbf{P} = \pi \\ &\sum_{v=1}^{N'} \pi(v) \quad = 1 \end{aligned} \tag{3}$$

In the final step of the DSPN solution algorithm the steady-state solution of the continuous-time DSPN, $p(.)$, is derived from the probability vector of the discrete-time embedded Markov chain, $\pi(.)$, and the conversion matrix C. Subsequently, the converted solution is normalized by Equation (5) in order to obtain the marking probability vector of the DSPN.

$$\tilde{p}(i) = \sum_{v=1}^{N'} C(v,i) \cdot \pi(v) \qquad 1 \le i \le N \tag{4}$$

$$p(i) = \frac{\tilde{p}(i)}{\sum_{v=1}^{N} \tilde{p}(i)} \qquad 1 \le i \le N \tag{5}$$

To illustrate the numerical DSPN solution algorithm an example of a for a single-server queueing system with constant service time and a finite capacity of K customers is considered. Figure 1 depicts a DSPN model of this queueing system. If free buffers are available, customers arrive according to an Erlang distribution with r phases. In case all buffers are occupied, the arrival process is blocked. Subsequently, this queueing system is referred to as the *modified $E_r/D/1/K$ queue*. The exponential transitions T2 and T3 are associated with a firing rate of $r\lambda$. The output arc from transition t1 to place P2 and the input arc from place P3 to transition T3 have the multiplicity r-1. Tokens contained in place P5 represent customers waiting in the queue or currently being served. The constant service requirement is modeled by the deterministic transition T4 with a firing delay of τ. The DSPN model has rK+1 tangible markings which can be classified as follows. One marking enables the deterministic transition exclusively, r markings enable only exponential transitions,

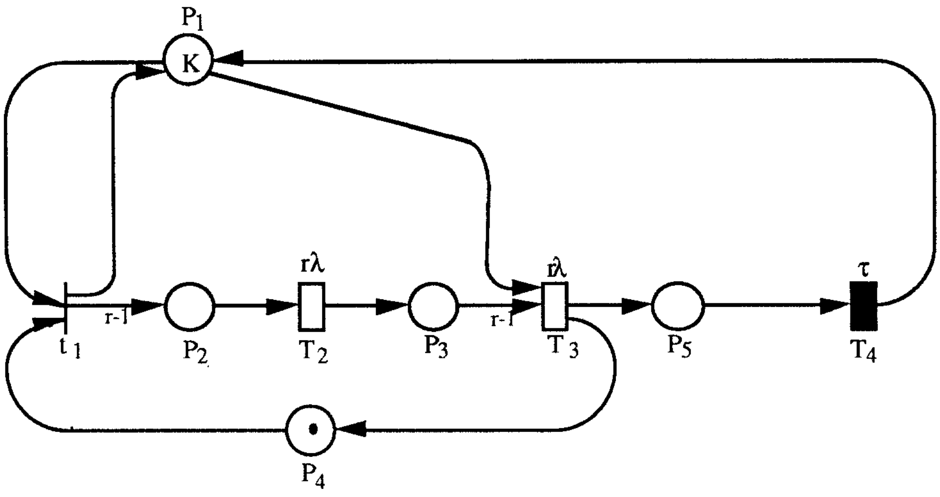


Figure 1. DSPN of the modified $E_r/D/1/K$ queue

and $r(K-1)$ markings enable one exponential transition concurrently with a deterministic transition. Observing the stochastic behavior of the modified $E_r/D/1/K$ queue only at departure points of customers defines an embedded Markov chain. Its state transition diagram is shown in Figure 2. The first index of a state $S_{i,j}$ denotes the number of customers in the system. The second index denotes the number of phases already completed by the next arriving customer. The states $S_{0,j}$ $(1 \le j \le r-1)$ have a single state transition with probability 1. In case of $1 \le i \le K-1$, each state $S_{i,j}$ has nonzero transition probabilities to the states $S_{i-1,j} \ldots S_{K-1,0}$. Note, only for the states $S_{1,0}$ and $S_{K-1,0}$ of this class are all feasible state transitions included in Figure 2. Since the continuous-time behavior of the modified $E_r/D/1/K$ queue is observed only at departure points, the embedded Markov chain consists of $r(K-1)+1$ states rather than $rK+1$.

In case the deterministic transition T4 is enabled the corresponding transition probabilities of the EMC and the entries of the conversion matrix are derived by a transient analysis of the Markov chain subordinated to this deterministic transition. This Markov chain is shown in Figure 3. In the first r rows only diagonal elements of the matrix C have nonzero values, since in the corresponding markings only exponential transitions are enabled. For the remaining rows of the conversion matrix C the coefficients are determined by the mean sojourn times in states of the subordinated Markov chain of transition T4 during the time interval $[0, \tau]$.

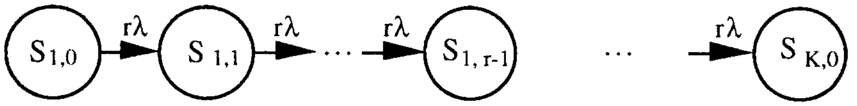


Figure 3. Subordinated Markov chain of the deterministic transition T4

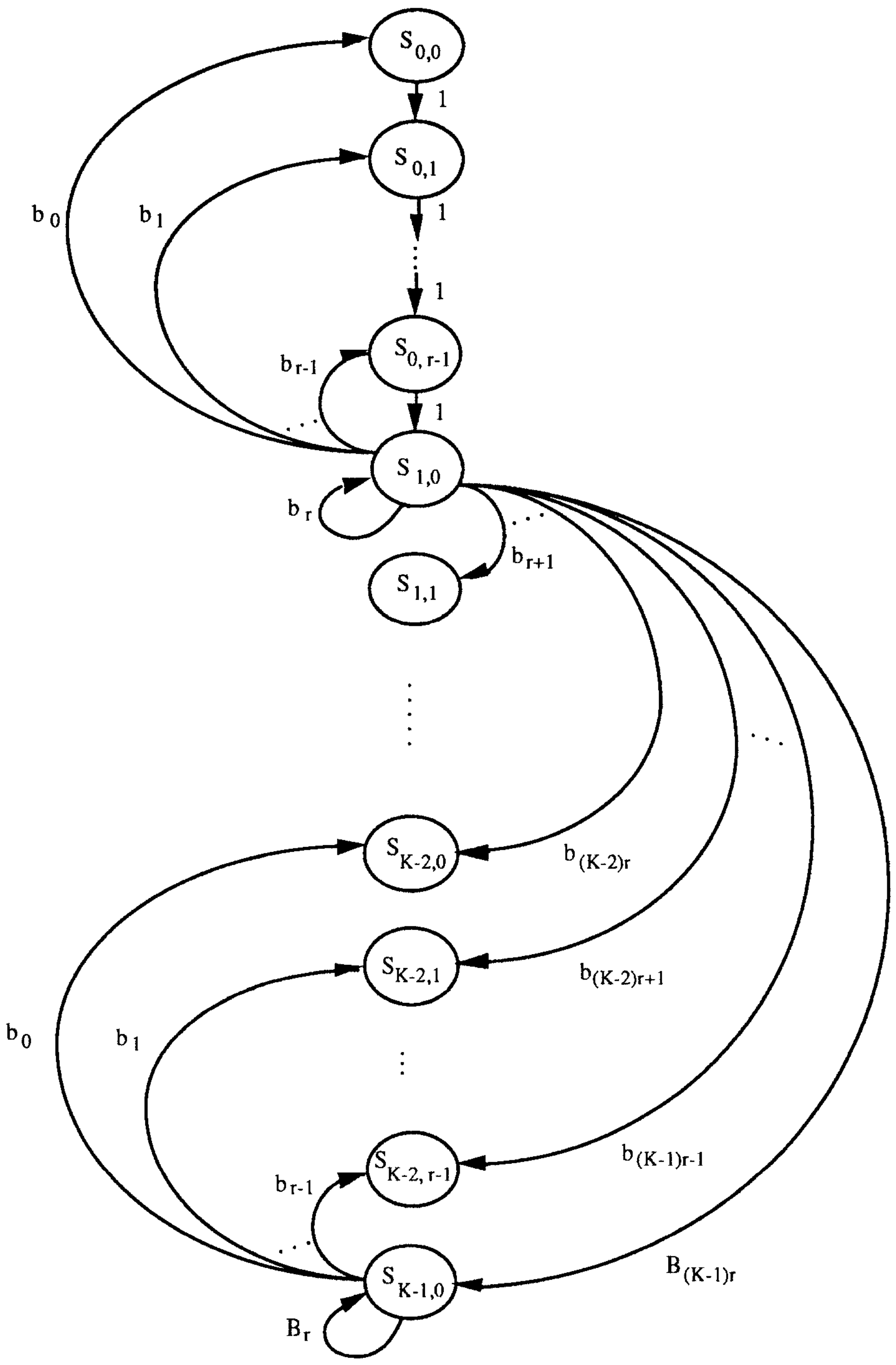


Figure 2. The embedded Markov chain of the modified E_r/D/1/K queue

The states $S_{K-1,1}$ ... $S_{K-1,r-1}$ and $S_{K,0}$ are not visited in the embedded Markov chain because the system is observed only at departure points of customers. In the continuous-time stochastic behavior of the modified $E_r/D/1/K$ queue these states may occur. The steady-state probabilities in the continuous-time stochastic behavior of the modified $E_r/D/1/K$ queue are derived by means of conversion factors. For this example, the conversion matrix C is a rectangular matrix of dimension $(r(K-1)+1) \times (rK+1)$. Thus, the conversion matrix is given by:

$$
C = \begin{bmatrix}
\frac{1}{r\lambda} & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\
0 & \frac{1}{r\lambda} & 0 & \ddots & & & & & & & & & & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots & & & & & & & & & \vdots \\
\vdots & & \ddots & \frac{1}{r\lambda} & 0 & 0 & & & & & & & & \vdots \\
0 & \cdots & \cdots & 0 & \frac{1}{r\lambda} & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\
0 & \cdots & \cdots & \cdots & 0 & c_0 & c_1 & c_2 & \cdots & \cdots & \cdots & & \cdots c_{(K-1)r-1} & c_{(K-1)r} \\
0 & \cdots & \cdots & \cdots & \cdots & 0 & c_0 & c_1 & \cdots & \cdots & \cdots & & \cdots c_{(K-1)r-2} & c_{(K-1)r-1} \\
\vdots & & & & & & \ddots & \ddots & \ddots & & & & \vdots & \vdots \\
\vdots & & & & & & & \ddots & \ddots & \ddots & & & \vdots & \vdots \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & c_0 & c_1 & \cdots & c_{r-1} & c_r
\end{bmatrix}
$$

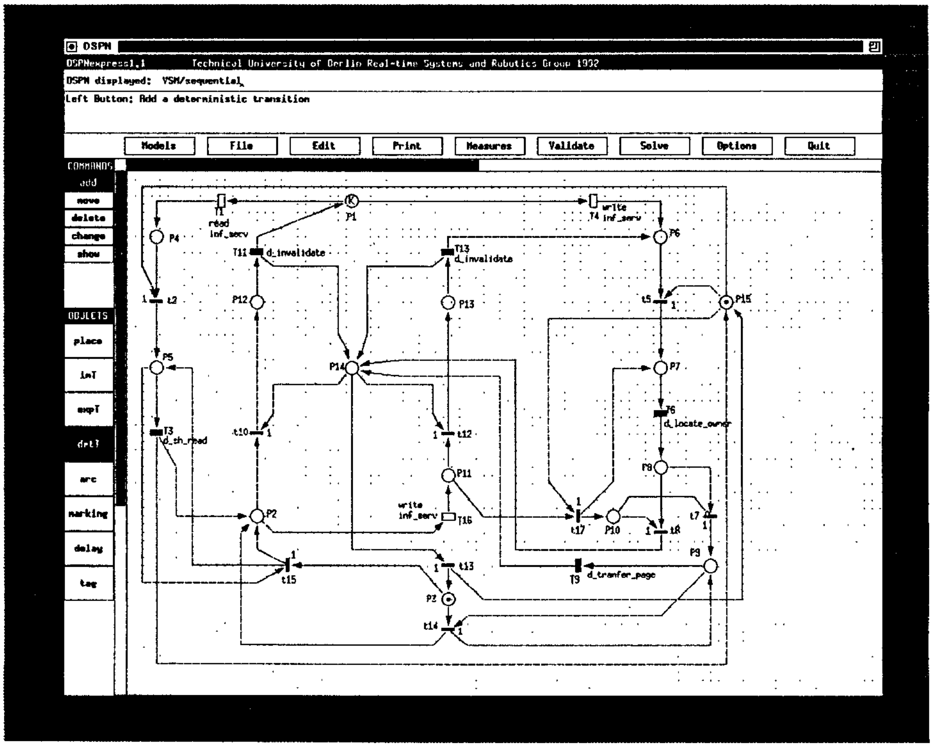
3 The Software Package DSPNexpress

The popularity of Petri Nets in which transition firings are augmented with time has gained from the availability of appropriate software packages which completely automate their solution process. This section gives a description of the analysis tool DSPNexpress which has been developed by the author and a group of students at the Technische Universität Berlin since 1991. The development of DSPNexpress has been motivated by the lack of a powerful software package for the numerical solution of DSPNs. The software architecture of DSPNexpress is particularly tailored to the numerical evaluation of DSPNs. Furthermore, DSPNexpress contains a graphical interface running under the X11 window system. A detailed description of DSPNexpress is given in [13].

In the following the main features of the software package DSPNexpress are outlined. The organization of DSPNexpress exploits the property that each Generalized Stochastic Petri Nets (GSPN) can be considered as a DSPN without deterministic transitions. As a consequence, a unified solution process for both DSPN and GSPN models is provided by DSPNexpress. The package contains an efficient numerical algorithm for calculating the transition probabilities of the EMC of a DSPN and the corresponding conversion factors. A similar algorithm is employed for calculating transient solutions of a GSPN. The DSPN solution module of DSPNexpress considers each connected component of a Markov chain subordinated to a deterministic transition of a DSPN, separately, for calculating the corresponding transition probabilities of the EMC and the conversion factors. This leads to a considerable reduction of the computational effort and the memory requirements of the DSPN solution algorithm. The separate transient evaluation of each connected component of a SMC is related to the decomposition approach on the net level proposed by Ajmone Marsan, Chiola, and Fumagalli [4]. In their decomposition approach, *DSPN subnets with independent behavior* have to be identified by means of structural analysis on the net level. To obtain the transition probabilities of the EMC of a DSPN this approach requires a proper combination of the transient quantities calculated separately for each subnet. The algorithm implemented in DSPNexpress employs a depth-first-search algorithm for deriving the generator matrices of connected components of each SMC from the reachability graph of tangible markings of a DSPN. The transient analysis of a SMC yields immediately the corresponding transition probabilities of the EMC underlying the DSPN. The interaction between software modules of DSPNexpress is performed mostly by interprocess communication by means of sockets. As a consequence, the system overhead required by reading from or writing to files is substantially reduced. Moreover, this allows a parallel execution of the transient analysis of SMCs on a cluster of workstations. Due to this efficient numerical DSPN solution algorithm DSPNexpress is able to calculate steady-state solutions of complex DSPNs with reasonable computational effort on a modern workstation. To the best of the author's knowledge DSPNexpress is the first software package with this feature.

The package DSPNexpress is organized as several sets of software modules which are stored in separate directories of a UNIX file system. We originally developed the package DSPNexpress for Sun™ workstations under SunOS.4.1, but the package has recently been ported to DEC™ and HP™ workstations under ULTRIX4.2 and HP-UX9.0, respectively. All software modules of DSPNexpress are implemented in the programming language C. To exploit the power of the numerical DSPN solution algorithm of DSPNexpress for solving complex DSPNs the package should run on machines with at least 16 MByte main memory. DSPNexpress allows a multi-user mode by including a link to the global directory DSPNexpress1.1 in the path expression in each user's shell profile. Only the model descriptions and the user-defined settings for DSPNexpress are stored in a local directory at each user's account.

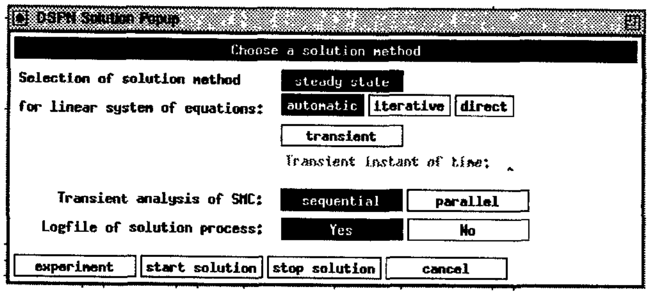
The graphical interface of DSPNexpress runs under the release 5 of the X11 window system and is implemented using athena widgets of the X11 programming library. It allows a user-friendly definition, modification, and quantitative analysis of DSPN models. Places (*place*), immediate transitions (*imT*), exponential transitions (*expT*), deterministic transitions (*detT*), and arcs (*arc*) of a graphical description of a DSPN are processed by selecting the corresponding object and one of the commands *add*, *move*, *delete*, or *change* with the mouse. For example, in the setting depicted Figure 4 deterministic transitions may be inserted and the grid option is used to simplify the graphical editing. Marking parameters (*marking*), firing delays (*delay*), and tags (*tag*) associated with places and transitions are processed in a similar way. For each setting of the command line an online help is provided in the upper part of the graphical interface explaining the actions currently available. The message displayed in Figure 4 indicates that clicking the left mouse button inserts a deterministic transition. In Figure 4 a DSPN named *sequential* of the directory *VSM* is displayed. Each place and each transition of this DSPN is labeled with a tag (e.g. T1, t2,.., etc). Each timed transition is also labeled with a parameter specifying the mean value of its firing delay (e.g. write or d_locate_owner). The exponential transitions T1, T4, and T16 are also labeled with *inf.-serv.* to specify their enabling policy as infinite-server [1]. To illustrate the capabilities of DSPNexpress the DSPN solution popup is shown in Figure 5. In case the steady-state solution of a DSPN shall be computed the numerical method for solving the linear system of the global balance equations of its EMC may be chosen by the user by clicking in one of the toggles *automatic*, *iterative*, or *direct*. Moreover, a user may specify whether the transient analysis of the SMCs is performed sequentially on a single workstation (*sequential*) or in parallel on a cluster of workstations (*parallel*) and whether verbose output of the solution process is displayed and stored in a logfile. Transient solutions of GSPNs are computed by clicking in the toggle *transient* and specifying an instant of time. The design of the graphical interface has been influenced by the interface of the version 1.4 of the package GreatSPN [7]. Opposed to the graphical interface of GreatSPN


DSPNexpress1.1 Technical University of Berlin Real-time Systems and Robotics Group 1992

DSPN displayed: VSM/sequential

Left Button: Add a deterministic transition

Figure 4. User Interface of DSPNexpress



Choose a solution method

Selection of solution method for linear system of equations: [✓ steady state] [✓ automatic] iterative direct

transient

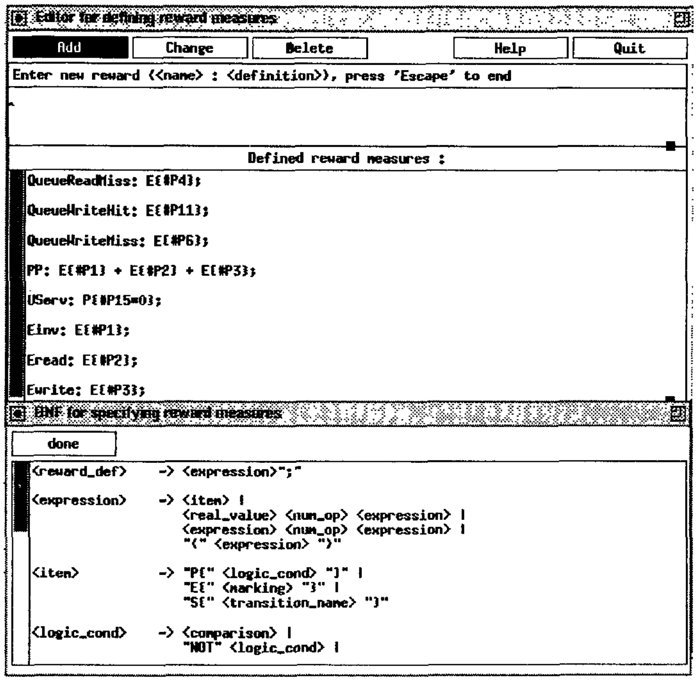
Transient instant of time:

Transient analysis of SMC: [✓ sequential] parallel

Logfile of solution process: [✓ Yes] No

Figure 5. DSPN solution popup

special purpose editors are provided by DSPNexpress for defining steady-state reward measures and marking-dependent firing delays. The editor for specifying user-defined reward measures is shown in Figure 6. The definitions of eight reward measures (QueueReadMiss, QueueWriteMiss, etc.) of the DSPN *sequential* are displayed in the middle part. The Backus-Naur form of the specification language for reward measures is displayed in the lower part of this window by clicking the help button. The graphical interface provides popups for changing firing delays of timed transitions, changing the type of a transition and for changing the multiplicity or the direction of an arc. Similar popups are also provided for defining or modifying a delay or a marking parameter, changing the string of a tag, etc.


Editor for defining reward measures
Add
Change
Delete
Help
Quit
BNF for specifying reward measures
done

Enter new reward (<name> : <definition>), press 'Escape' to end

Defined reward measures :

```
QueueReadMiss: E{#P4};
QueueWriteHit: E{#P11};
QueueWriteMiss: E{#P6};
PP: E{#P1} + E{#P2} + E{#P3};
UServ: P{#P15=0};
Einv: E{#P1};
Eread: E{#P2};
Ewrite: E{#P3};
```

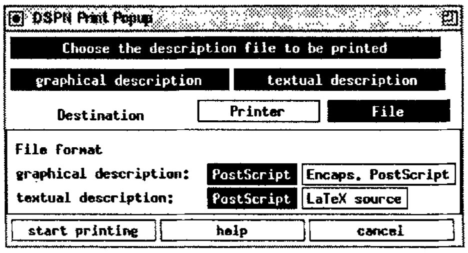
```
<reward_def>     -> <expression>";"

<expression>     -> <item> |
                    <real_value> <num_op> <expression> |
                    <expression> <num_op> <expression> |
                    "(" <expression> ")"

<item>           -> "P{" <logic_cond> "}" |
                    "E{" <marking> "}" |
                    "S{" <transition_name> "}"

<logic_cond>     -> <comparison> |
                    "NOT" <logic_cond> |
```

Figure 6. Reward measure editor



DSPN Print Popup

Choose the description file to be printed

[✓ graphical description] [✓ textual description]

Destination: Printer [✓ File]

File format

graphical description: [✓ PostScript] Encaps. PostScript

textual description: [✓ PostScript] LaTeX source

start printing | help | cancel

Figure 7. DSPN print popup

The print popup of DSPNexpress is shown in Figure 7. This popup allows the printing of the DSPN currently being displayed in the main window of DSPNexpress and/or its stochastic description such as delay and marking parameters and definitions and computed results of user-defined reward measures. These description may be either directly printed on a laser writer or stored in files named *<model>.graph* and *<model>.text*, respectively, by clicking the appropriate toggles with the left mouse button.

4 A DSPN of a Sequential Memory Consistency Protocol

Previous performance studies of cache consistency and memory consistency protocols of shared memory multiprocessors or multicomputer systems consider besides the read and write request rates to a shared block or page also the probabilities that a shared block reside in a particular state as input parameters of the model. As measure of interest the overall processing power has been derived from the steady-state solution of the analytical model or have been considered in the simulation experiments. Opposed to that the modeling approach presented in this section takes only the read and write request rates as input parameters. The long-run probabilities that a shared page resides in the states INVALID, SHARED, or EXCLUSIVE are derived from the steady-state solution of the DSPN.

Trace-driven simulation has been employed for evaluating the performance of consistency models for a multiprocessor system with shared-memory, but a trace-driven simulation study requires substantially more effort in computation time and memory space than an analytical model. Moreover, a simulation model (both trace-driven and stochastic) does usually not allow the verification of qualitative properties of consistency models. DSPNs are a numerically solvable modeling tool which are both suited for verification of qualitative properties of consistency paradigms and quantitative performance analysis. The quantitative results derived by a DSPN can be calculated with a pre-determined numerical accuracy. Due to the greater level of detail trace-driven simulation yield more accurate quantitative results for a specific application program than a DSPN.

A DSPN model of the behavior of a single shared page in a VSM system with sequential consistency and a write-invalidate protocol is depicted in Figure 8. A MIMD architecture is considered in which the nodes are connected by a low-latency scalable interconnection network (e.g. a crossbar interconnection network). The DSPN considers a subset of K+1 nodes of the multicomputer system which are competing for the same shared page. Each of the K tokens in place *INVALID* and the token in place *EXCLUSIVE* represent one of these nodes. The three main states of a shared page in the global address space, namely INVALID, SHARED and EXCLUSIVE are represented in the DSPN by places with corresponding labels. The token in place *Server available* represents the idle state of the server which maintains the data structures for managing the considered page. All other places and transitions of Figure 8 are also labeled according to their meaning. In the DSPN model requests causing a page status change from SHARED to EXCLUSIVE are given a higher priority than those from INVALID to EXCLUSIVE or from INVALID to SHARED. The latter ones are given the same priority. This is encoded in the DSPN model as follows. The immediate transition *Start process find owner* has associated a firing priority of 2 whereas the immediate transitions *Start process read miss* and *Start process write miss* have associated a firing priority of 1 and an equal weight. These assumptions can be easily modified by changing the appropriate firing priorities and weights of these immediate transitions. Since the delay required for a page status change is typically at least one order of magnitude smaller than a page transfer, this delay is neglected in the DSPN. In case of a read miss this approximation allows to represent a location of the owner and a page transfer by the single deterministic transition *Find owner and transfer page delay.*

The following explains the representation of the activities in the DSPN caused by a write miss to the shared page. Activities caused by the other types of requests are represented in a similar way. The firing of the exponential transition *Write miss* removes a token from the place INVALID and puts it to the place *Write miss queue.* In case the server is available (a token resides in place *Server available*) the immediate transition *Start process write miss* fires removing a token from the places *Server available* and *Write miss queue* and putting a token to the place *Process find owner.* The delay required by the system for this activity is represented by the deterministic transition *Find owner delay.* A firing of this transition puts the token in place *Decision.* Since the place *Write hit occurred* contains no token the immediate transition *Is write miss* fires and puts the token in the place *Process transfer page.* If the place *EXCLUSIVE* contains a token (the node owning the page has currently exclusive write permission), the immediate transition *Change access due to write miss* fires and puts this token in the place *SHARED.* This constitutes the first step of the invalidation of a former owner. The former owner is not stalled until a copy of the page has been transferred to the new owner, since residence times of tokens in the place *SHARED* contributes to the processing power. After the deterministic transition *Transfer page delay* has fired, the token residing in place *Process transfer page* is moved to the place *Process invalidations.* In case no further write request has occurred at the former owner (the exponential transition *Write hit* has not fired), the immediate transition *Start invalidating* fires and puts the token from place *SHARED* to the place *Invalidating.* This activity constitutes the second step of the invalidation

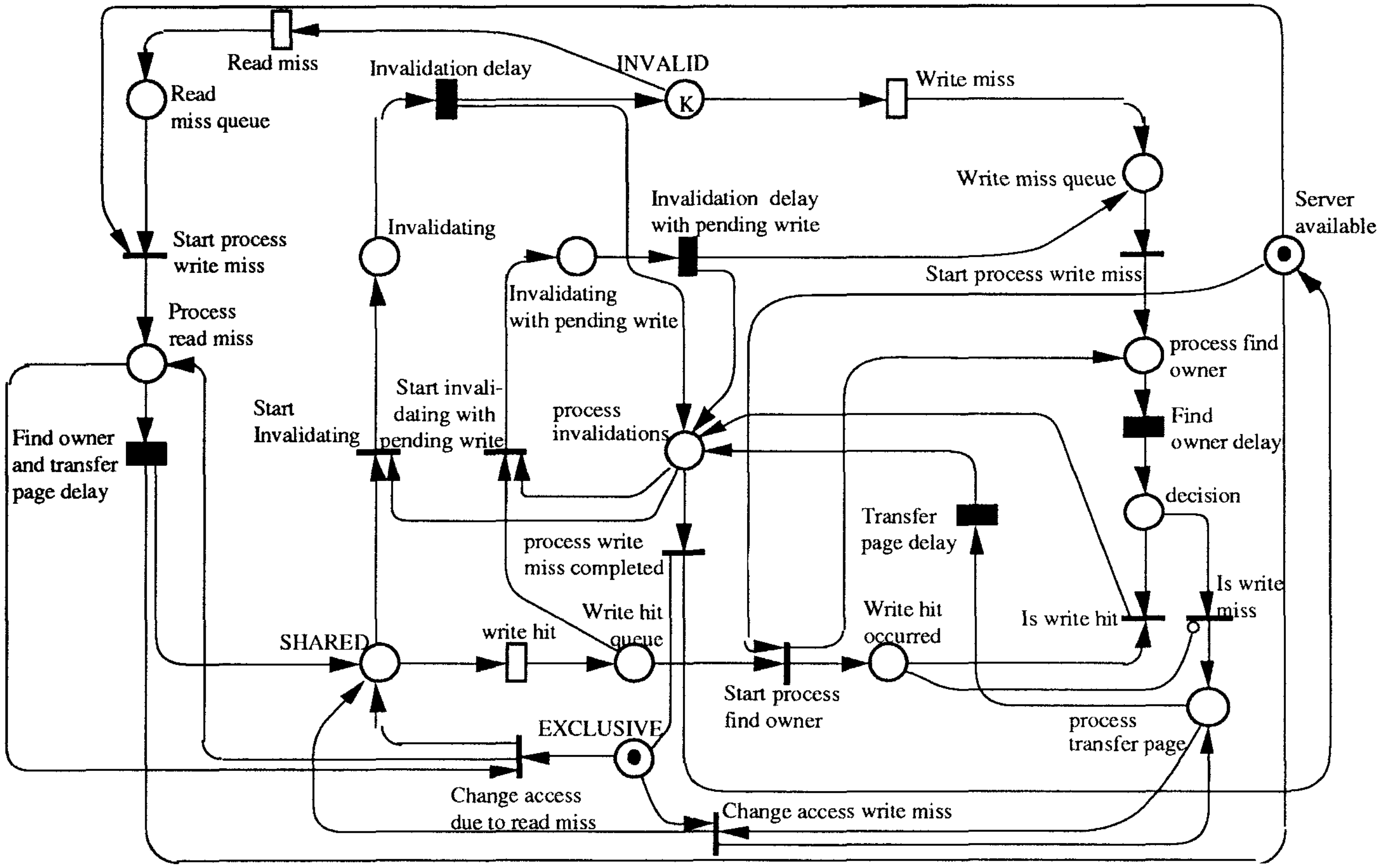


Figure 8. DSPN model of the behavior of a shared page in a VSM system with sequential consistency and a write-invalidate protocol [17]

which is equal to the invalidation of nodes with read permission. The firing of the deterministic transition *Invalidation delay* removes the token from place *Invalidating* and puts a token to the place *INVALID* and *Process Invalidations.*

Invalidations are modeled sequentially and represented by the subnet consisting of the places *Invalidating* and *Invalidating with pending write* together with the transitions *Start invalidation, Start invalidation with pending write, Invalidation delay*, and *Invalidation delay with pending write.* All nodes with SHARED access have to be invalidated before the new owner gets the access EXCLUSIVE. Thus, the immediate transitions *Start invalidation* and Start *invalidation with pending write* have associated the higher firing priority 2 than the immediate transition *Process write miss completed* which has priority 1. All other immediate transition have also associated the firing priority 1. The firing of the immediate transition *Process write request completed* removes the token from place *Process invalidations* and puts a token to each of the places *EXCLUSIVE* and *Server available.* Curves of the processing power achieved by a multicomputer system in which virtually shared memory is implemented by the sequential memory consistency model can be directly obtained from the steady state solution of the DSPN of Figure 8 and were presented in [17].

This DSPN model contains 5 deterministic transitions and its state space grows rather fast for increasing the number of tokens. Thus, for increasing marking parameter K the calculation of the solution of this DSPN is severely hampered due to state space explosion. For example, in case of K = 20 the DSPN has 59,101 tangible markings and its EMC consists of 6,004,585 nonzero state transitions. As shown in [14], DSPNs of this complexity could not be solved in practice with the adaptive matrix exponentiation method implemented in GreatSPN1.4. The experimental results presented in [13] show that this DSPN can be solved by DSPNexpress with reasonable computational effort on a modern workstation.

Recent Results

To summarize, the current state-of-the-art is that steady-state solutions of quite complex DSPNs can be calculated using DSPNexpress with reasonable computational effort on a workstation [13], [14]. This progress in the analysis of DSPNs has triggered other methodological work on DSPNs. Trivedi, Choi, and Mainkar introduced a technique for sensitivity analysis of DSPN [19]. In another recent work, extensions to the numerical solution method of DSPNs were introduced in order to cope with deterministic transitions with marking-dependent firing delays [16]. However, the numerical solution method for DSPNs which is currently implemented in the package DSPNexpress is still based on the restriction that in no marking two or more deterministic transitions are concurrently enabled. This structural restriction of DSPNs severely hampers their applicability for a wide gamut of problems. In order to fully establish DSPNs in the research community numerical methods for transient analysis and steady-state analysis of DSPNs with concurrently enabled deterministic transitions have to be developed. Recently, Choi, Kulkarni, and

Trivedi showed how transient solutions of DSPNs can be computed using numerical inversion of a Laplace-Stieltjes transform [8]. The same authors showed that the stochastic process underlying a DSPN is a Markov regerative stochastic process and introduced an approach for dealing with non-exponentially distributed firing delays other than the deterministic distribution [9]. However, in all of their work the structural restriction that in no marking of the DSPN two or more deterministic transitions are concurrently enabled is still assumed.

To relax this restriction the employment of the method of supplementary variables has been proposed for analyzing DSPNs [10]. In this paper it has been shown that the computation of steady-state solutions of DSPNs with concurrently enabled deterministic transitions by the proposed solution approach requires the solution of a partial system of differential equations. Furthermore, DSPNs in which the firing delay of timed transitions is either exponentially or non-exponentially distributed, but no concurrent firings of transitions with nonexponential delay have been considered. Employing the proposed solution approach for such DSPNs leads to a system of ordinary differential equations and integral equations. In case the non-exponential distributions belong to the class of polynomial distributions efficient computational formulas for solving this system of integro-differential system have been introduced by extending Jensen's method, also called randomization or uniformization [11].

References

[1] M. Ajmone Marsan, G. Balbo, A. Bobbio, G. Chiola, G. Conte, and A. Cumani. The Effect of Execution Policies on the Semantics of Stochastic Petri Nets. *IEEE Trans. Softw. Engin.*, 15, pages 832-846, 1989.

[2] M. Ajmone Marsan and G. Chiola. On Petri Nets with Deterministic and Exponentially Distributed Firing Times", in: *G. Rozenberg (Ed.) Advances in Petri Nets 1986, Lecture Notes in Computer Science 266*, pages 132-145, Springer 1987.

[3] M. Ajmone Marsan, G. Chiola, and A. Fumagalli. An Accurate Performance Model of CSMA/CD Bus LAN, in: *G. Rozenberg (Ed.) Advances in Petri Nets 1986, Lecture Notes in Computer Science 266* pages 146-161, Springer 1987.

[4] M. Ajmone Marsan, G. Chiola, and A. Fumagalli. Improving the Efficiency of the Analysis of DSPN Models. in: *G. Rozenberg (Ed.) Advances in Petri Nets 1989, Lecture Notes in Computer Science 424*, pages 30-50, Springer 1990.

[5] M. Ajmone Marsan and V. Signore. Timed Petri Net Performance Models for Fiber Optic LAN Architectures, *Proc. 2nd Int. Workshop on Petri Nets and Performance Models, Madison Wisconsin*, pages 66-74, 1987.

[6] G. Balbo, G. Chiola, G. Franceschinis, and G. Molinar Roet. On the Efficient Construction of the Tangible Reachability Graph of Generalized Stochastic Petri Nets. *Proc. 2nd Int. Workshop on Petri Nets and Performance Models, Madison Wisconsin* , pages 85-92, 1987.

[7] G. Chiola. A Graphical Petri Net Tool for Performance Analysis. *Proc. 3rd Int. Workshop on Modelling Techniques and Performance Evaluation, Paris France*, pages 323-333, 1987.

[8] H. Choi, V. Kulkarni, and K.S. Trivedi. Transient Analysis of Deterministic and Stochastic Petri Nets, *Proc. 14th Int. Conf. on Applications and Theory of Petri Nets, Chicago Illinois, June 1993* (to appear).

[9] H. Choi, V. Kulkarni, and K.S. Trivedi. Markov Regenerative Stochastic Petri Nets, *Proc. 16th Int. Symposium on Computer Performance Modeling, Measurement, and Evaluation (PERFORMANCE '93), Roma Italy October 1993*, (to appear).

[10] R. German and C. Lindemann. Analysis of Stochastic Petri Nets by the Method of Supplementary Variables, *Proc. 16th Int. Symposium on Computer Performance Modeling, Measurement, and Evaluation (PERFORMANCE '93), Roma Italy October 1993*, (to appear).

[11] D. Gross and D.R. Miller. The Randomization Technique as a Modeling Tool and Solution Procedure for Transient Markov Processes, *Operations Research, 32*, pages 345-361, 1984.

[12] M.A. Holliday and M.K. Vernon. A Generalized Timed Petri Net Model for Performance Analysis, *IEEE Trans. Softw. Engin., 12*, pages 1297-1310, 1987.

[13] C. Lindemann. DSPNexpress: A Software Package for the Efficient Solution of Deterministic and Stochastic Petri Nets, *Proc. 6th Int. Conf. on Modeling Techniques and Tools for Computer Performance Evaluation, Edinburgh Scotland*, pages 15-29, 1992.

[14] C. Lindemann. An Improved Numerical Algorithm for Calculating Steady-state Solutions of Deterministic and Stochastic Petri Net Models, *Performance Evaluation, 18*, 1993 (in press).

[15] C. Lindemann, G. Ciardo, R. German, and G. Hommel. Performability Evaluation of an Automated Manufacturing System with Deterministic and Stochastic Petri Nets, *Proc. Int. Conf on Robotics and Automation, Atlanta Georgia*, pages 576-581, 1993.

[16] C. Lindemann and R. German. Modeling Discrete Event Systems with State-dependent Deterministic Service Times, *Discrete Event Dynamic Systems: Theory and Applications, 3*, pages 249-270, 1993.

[17] C. Lindemann and F. Schön. Evaluating Sequential Consistency in a Virtually Shared Memory System with Deterministic and Stochastic Petri Nets; *Proc. Int. Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, San Diego California*, pages 63-68, 1993.

[18] M. Lu, D. Zhang, and T. Murata. Analysis of Self-Stabilizing Clock Synchronization by Means of Stochastic Petri Nets, *IEEE Trans. on Computers, 39*, pages 597-604, 1990.

[19] K.S. Trivedi, H. Choi, and V. Mainkar. Sensitivity Analysis of Deterministic and Stochastic Petri Nets, *Proc. Int. Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, San Diego California*, pages 271-276, 1993.

[20] W.M. Zuberek. Timed Petri Nets: Definitions, Properties, and Applications, *Microelectronics & Reliability, 31*, pages 627-644, 1991.