Performability Evaluation of Grid Environments using Stochastic Reward Nets

Reza Entezari-Maleki, *Computer Engineering Department, Sharif University of Technology, Iran,*
Kishor S. Trivedi, *Electrical and Computer Engineering Department, Duke University, NC, US,*
and Ali Movaghar, *Computer Engineering Department, Sharif University of Technology, Iran*

Abstract—In this paper, performance of grid computing environment is studies in the presence of failure-repair of the resources. To achieve this, in the first step, each of the grid resource is individually modeled using Stochastic Reward Nets (SRNs), and mean response time of the resource for grid tasks is computed as a performance measure. In individual models, three different scheduling schemes called random selection, non-preemptive priority, and preemptive priority are considered to simultaneously schedule local and grid tasks to the processors of a single resource. In the next step, single resource models are combined to shape an entire grid environment. Since the number of the resources in a large-scale grid environment is more than can be handled using such a monolithic SRN, two approximate SRN models using folding and fixed-point techniques are proposed to evaluate the performance of the whole grid environment. Brouwer's fixed-point theorem is used to theoretically prove the existence of a solution to the fixed-point approximate model. Numerical results indicate an improvement of several orders of magnitude in the model state space reduction without a significant loss of accuracy.

Index Terms—Performance, availability, grid environment, mean response time, stochastic reward net, Markov reward model, Markov chain

1 Introduction

Grid computing environment is composed of many resources distributed within multiple virtual organizations and administrative domains [1], [2]. Grid enables coordinated resource sharing and problem solving in dynamic and multi-institutional organizations [1]. Computational grids have been found to be very powerful environments to solve the computational- and data-intensive problems in science and industry. To use the tremendous capabilities of the grid computing environment, grid users should deliver their own tasks to the environment. To do this, grid users interact with Resource Management System (RMS) or Grid Manager (GM) to submit the tasks to the environment. Afterward, RMS dispatches the tasks to the distributed resources within the environment. Whenever a grid resource obtains a grid task, it starts to service the task, and then, returns results to the users [2]–[5].

In order to deliver the tasks to the grid environment, distributed resources should be available to interact with RMS and grid users. Processors existing inside a resource can fail to execute a grid task at any time. Therefore, the availability of a grid resource to service grid tasks can be highly influenced by processor failure. It is worthwhile to mention that although RMS can fail during servicing the grid tasks, basically it is considered to be a very powerful and failure free resource in the environment [2]–[6]. On the other hand, grid resources should service local tasks submitted to them directly by local users in their administrative domains. When a user existing in a resource's administrative domain submits a task to the resource, the task is serviced inside that resource along with the grid tasks submitted by grid users [1], [7], [8]. Based on the processor management policies existing inside a resource, different scheduling schemes can be considered to simultaneously dispatch grid and local tasks among processors of the resource. In some cases, a higher execution priority is assigned to service local tasks with respect to the grid tasks, but generally, any kind of scheduling can be applied.

In addition to the grid resource availability which influences user perception of grid environment usefulness, performance of the grid is also considered as one of the most important user satisfaction factors. Generally, performance is the key issue for any system that services user requests and can be defined considering the specification of the system and user expectations. In grid computing environment, performance evaluation mainly focuses on the time to service the tasks submitted by grid and local users. In this respect, several related measures such as mean response time and mean waiting time of

- *R. Entezari-Maleki is with both the Department of Computer Engineering, Sharif University of Technology, Tehran, Iran and the Department of Electrical and Computer Engineering, Duke University, Durham, NC, 27708.*
  E-mail: entezari@ce.sharif.edu and re50@duke.edu
- *K.S. Trivedi is with the Department of Electrical and Computer Engineering, Duke University, Durham, NC, 27708,*
- *A. Movaghar is with the Department of Computer Engineering, Sharif University of Technology, Tehran, Iran.*

the grid tasks can be defined. Moreover, blocking probability of grid task arrivals can be one of the interesting measures in this kind of distributed computing systems in which designing a system with low blocking probability is important [3], [4], [9]–[11]. In traditional performance evaluation, each of these measures is assessed without any consideration of availability/reliability of a resource. Nevertheless, in highly distributed computing systems consisting of many independent resources (e.g. grid computing environments) each of the resources can fail or even can be added to or removed from the system at any time. Therefore, the performance of a single resource and whole distributed system is highly dependent on the number and processing power of the existing resources and their processors.

Therefore, analyzing the pure performance of a grid resource, and consequently grid environment tends to be optimistic since it ignores the failure-repair behavior of the processors and resources. On the other hand, pure availability analysis tends to be conservative, since performance considerations are not taken into account. As a result, combined performance and availability, called performability [12], [13], evaluation of the grid computing environment can present most realistic view of grid system behavior, and help to appropriately compute the mean response time of the system for grid tasks. To achieve this, we use Stochastic Reward Nets (SRNs) to model and evaluate the performability of a single grid resource and the whole grid environment, in this paper. SRN is an extension of Stochastic Petri Nets (SPNs) which has the advantage of specifying and evaluating a real system in a compact and intuitive way. SRN has emerged as a powerful modeling paradigm in performance, availability and reliability analysis of fault tolerant computing and communication systems [14]–[23] as it enables the automated generation and solution of large Markov reward models.

The proposed method is discussed in two steps. In the first step, we model a single grid resource using SRNs and compute mean response time of the resource to grid tasks. Three different SRNs are proposed to model a single grid resource considering three different scheduling schemes applied to schedule grid and local tasks to the processors inside a resource. The first scheduling scheme does not consider any priority between grid and local tasks, but the next two schemes consider non-preemptive and preemptive priorities for local tasks over the grid ones. After modeling single grid resources in the first step, the SRN models of the individual single resources are combined and an entire grid environment is captured in the second step. Since, the number of the resources in real grid environments is rather large, the combined model encounters the state space explosion problem in which the number of the states in the underlying Markov chain of the SRN model becomes more than can be handled by existing tools. Therefore, the SRN resulting from combining SRN models of single resources is not scalable, and cannot be used to model and evaluate the performability of real grid environments. To overcome this problem, two approximate SRN models are proposed. The first approximate model uses folding technique to model whole grid environment and the second one uses decomposition approach and fixed-point iteration method [20], [24], [25] to solve the sub-models. The existence of a fixed-point is proven using Brouwer's fixed-point theorem. Moreover, some simple examples are presented to show how the models can be applied to evaluate the performability of a single grid resource and that of a grid environment. The example provided for modeling and evaluating entire grid environment obviously shows that the approximate SRN models considerably decrease the number of the states in the underlying Markov chain of the SRNs without any undue loss of accuracy and can be used for real systems.

Main contribution of this paper is modeling a single grid resource to compute the combined performance and availability measure of the resource. We model different scheduling schemes to simultaneously schedule grid and local tasks to the processors of a resource. Moreover, presenting two models to approximate the non-scalable monolithic model of an entire grid environment to capture a real grid is another contribution of the paper. The main advantage of the approximate models presented in this paper is their capability in modeling large scale grids without showing any state space explosion.

The rest of the paper is organized as follows. Section 2 introduces some related research done in the field of grid performance and dependability evaluation. Moreover, some research papers related to the concept of modeling various computing systems by SRNs which use approximate models are introduced in this section. In Section 3 and Section 4, the proposed models for combined performance and availability analysis of a single grid resource and whole grid environment are presented, respectively. Furthermore, in both Section 3 and Section 4, detailed examples together with numerical results are provided. Finally, Section 5 concludes the paper and presents future work.

2 Related Work

Many analytical models have been proposed to evaluate the performance and dependability measures of various computing systems. Some of the models evaluate performance and dependability measures separately and some others simultaneously take both of them into account. In the following, some of the related work in this research area especially on grid and distributed computing systems are introduced.

Azgomi et al. [6] have presented a Colored Petri Net (CPN) model to show the workflow of task execution in grid computing, and compute the reliability of a grid service. Although the CPN proposed in [6] precisely investigates the failure event of grid resources, it ignores considering local tasks of grid resources. Entezari-Maleki et al. [10] have proposed a Markov chain model to compute total makespan and mean response time for a single task in grid computing environments. The main advantage of the model proposed in [10] is that it considers more than one manager for the grid environment. Grid managers are distributed within the environment and they collaborate with each other to solve a task submitted to the environment by grid user. However, the model presented in [10] still ignores the failure-repair event of grid resources and the performance is evaluated purely without any consideration about dependability issues. Parsa et al. [11], [26] have proposed queuing network and Generalized Stochastic Petri Net (GSPN) solutions to model and evaluate performance of grid computing systems. Both queuing network and GSPN models proposed in [11] only consider grid tasks submitted by grid users paying no attention to the local tasks of the system, whereas the models proposed in [26] consider both types of tasks, grid and local tasks. Difficulty with all of the models proposed in [26] and [11] is that the models can only compute pure performance of the grid environment and they do not handle the situation in which one or more of the resources fail.

Longo et al. [22] have proposed an SRN model to analyze the availability of large scale IaaS cloud. The model is firstly presented in its monolithic form in one level, and then, decomposed to overcome the largeness problem caused by the monolithic model. Dependencies among sub-models of the decomposed model are found, and then, fixed-point iteration method is used to solve the interacting sub-models. Another model in the same context has been proposed in [23] by Ghosh et al. In [23], interacting stochastic sub-models were presented to evaluate the performance of large scale IaaS clouds while workload of the cloud, system characteristics and management policies are taken into consideration. After using fixed-point iteration to solve interacting sub-models, two performance measures, mean response time and job rejection probabilities, were computed.

Ibe et al. [27] have used GSPNs to model polling systems and compute mean response time of such a system in different cases. Single service, Finite population and finite capacity extensions of a polling system were considered in [27]. Based on the models presented in [27], Choi et al. [17] have proposed approximate performance models for polling systems which can be used to analyze the performance of such systems when the number of nodes and their potential customers are large. A decomposition approach was used to solve the problem, and two sub-models were developed to interact with each other to solve the overall model. Fixed-point iteration method was used to solve interacting sub-models.

Tomek et al. [18] have proposed a Markov chain based model to analyze the availability of a large fault tolerant system with shared repair facility. Moreover, an SRN has been proposed to model the aforementioned system. Fixed-point iteration method has been used to solve the problem whenever the number of systems becomes larger than can be managed by monolithic models. SRNs and decomposition approach to solve interacting models in the context of channel allocation in wireless networks have been used by Ma et al. [19]. The SRNs presented in [19] appropriately model new and hand-off calls in wireless networks and evaluate call dropping and blocking probabilities along with call waiting times as performance measures of such systems. Mainkar et al. [20] have proposed an SRN to model and analyze the performance of heterogeneous multiprocessor systems. The model considers non-preemptive priority scheme between tasks and tries to solve the model by generating underlying Markov chain of the proposed SRN. After proposing a monolithic SRN model for an assumed multiprocessor system, an approximate model with two interacting sub-models were presented to solve the problem in the general case. Fixed-point iteration method was exploited to solve the interacting sub-models and compute performance measure in the form of utilization of a processor.

Other related research in this field also can be found in the literature. Generally, each of the methods presented in this research area has its own pros and cons. On one side, the main problem existing in previously proposed methods in grid context is that only a few of them consider both local and grid tasks, and simultaneous execution of them inside grid resources. Some papers that consider both types of tasks only compute pure performance of the grid environment ignoring failure-repair behavior of the resources. Moreover, studying a grid resource collaborating with other resources in the environment and considering the effect of a resource on the performance of other resources while there are some shared facilities (e.g. shared grid queue) are still open issues. On the other side, models proposed for other systems are not suitable to be directly applied to grid environments, because they do not consider the specific characteristics of grids. Each of the models has been proposed to solve a given system with specific input parameters (system requirements) and particular output metrics (performance or dependability measures) in which both input and output parameters concretely depend on the system under study.

3 SRN Models for Single Grid Resource

Three SRN models are presented in this paper to evaluate the combined performance and availability of a single grid resource. It should be mentioned that we do not present the concept of SRNs here because of the lack of the space, but for more information about Petri nets (PNs), timed and stochastic extensions of PNs, and finally SRNs please see [25], [28]–[31].

In this paper, three scheduling schemes for simultaneous execution of grid and local tasks are considered. There are other scheduling schemes which can be modeled using SRNs [32], [33], but the aim of this section is only showing the possibility of modeling different scheduling schemes using the proposed SRN models for single resources, and then, applying our proposed methods for combining the single models to capture the entire model. In the first scheme, there is no priority between grid and local tasks. So, if both tasks request the resource, one of them is selected randomly to be assigned to an idle processor inside that resource. In the second and third models, a higher priority is assigned to local tasks over the grid tasks. So, a grid task is executed by one of the idle processors of the grid resource only if there is no local task in local queue of that resource. The difference between two later models is that the second model considers non-preemptive priority scheme in which a local task cannot preempt a running grid task, but the third model applies preemptive priority. In preemptive priority scheme, when a local task arrives to the resource and finds all the processors busy, the running grid task is preempted and the processor is given to the newly arrived local task. The preempted grid task is returned to the grid queue to be assigned to one of the idle processors later. In the following, all three models are discussed in details.

3.1 First Model

The first model is shown in Fig. 1. Input parameters of this model are: (1) grid and local queue sizes of the resource (M_G and M_L), (2) grid and local tasks arrival rates (λ_G and λ_L), (3) number of the processors inside the resource (N), (4) service rate of each processor (μ), (5) failure rate of idle and busy processors represented by γ_i and γ_b, respectively which $\gamma_i < \gamma_b$, and (6) repair rate of a failed processor (δ). It should be mentioned that the times assigned to all timed transitions follow exponential distribution.

As mentioned earlier, the aim of the models is to evaluate the performance of a grid resource while availability of the resource is taken into account. To do this, the failure-repair behavior of the processors in a grid resource should be considered in the models. Places P_{UP} and P_{DP} represent the up and down processors inside the resource, respectively. It is assumed that there are N operational homogeneous processors

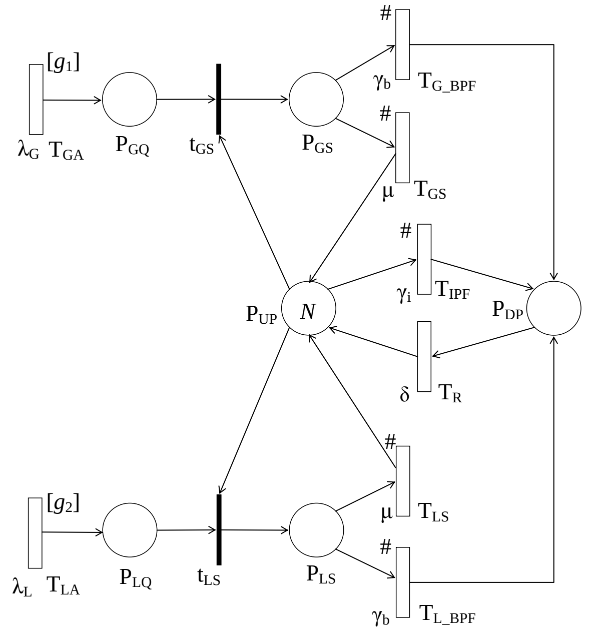

Fig. 1. First SRN Model of a Single Grid Resource (Without Priority)

TABLE 1
Guard Functions of SRN Model Shown in Fig. 1

Guard Functions	Values
g_1	1 if $[\#P_{GQ}] < M_G$
	0 otherwise
g_2	1 if $[\#P_{LQ}] < M_L$
	0 otherwise

inside a resource when the resource starts to service the tasks. Transition T_{IPF} represents the failure event of idle processors. The pound sign (#) in the arc from the place P_{UP} to the transition T_{IPF} shows that the firing rate of this transition is marking dependent. So, the actual firing rate of transition T_{IPF} is computed as $k.\gamma_i$, where k is the number of tokens in place P_{UP}. Transition T_R represents the repair process of a failed processor. It is assumed that there exists only one repairperson for all processors of a single resource.

As can be seen in Fig. 1, there are two different lines of tasks' arrivals. The first line is for grid tasks submitted by grid users and the second line is for local tasks which are submitted by local users inside administrative domain of the resource. Transitions T_{GA} and T_{LA} show the arrival of grid and local tasks to the resource, respectively. There are two guard functions in the model associated with timed transitions T_{GA} and T_{LA} to reflect the grid and local queue sizes of the resource. These functions are described in Table 1, where M_G and M_L denote the sizes of grid and local queues, respectively. Once transition T_{GA} (T_{LA}) fires, a token is deposited in place P_{GQ} (P_{LQ}) showing a

TABLE 2
New Guard Function of the Second Model

Guard Functions	Values
g_3	1 if $[\#P_{LQ}] = 0$
	0 otherwise

TABLE 3
New Guard Function of SRN Model Shown in Fig. 2

Guard Functions	Values
g_4	1 if $[\#P_{UP}] = 0$
	0 otherwise

grid (local) task has been submitted to the resource and it is waiting to get service from the resource.

If there is a token in place P_{GQ} (P_{LQ}) and there is at least one token in place P_{UP}, one token from P_{GQ} (P_{LQ}) together with another token from P_{UP} is removed and a token is put in the place P_{GS} (P_{LS}) representing a grid (local) task is getting service from the resource. It is clear that there is no priority between grid and local tasks in the model shown in Fig. 1. Therefore, if both immediate transitions t_{GS} and t_{LS} are enabled, one of them will fire randomly with equal probability, and an idle processor will be assigned to a grid or local task. Firing one of the immediate transitions t_{GS} or t_{LS} may cause another transition to be disabled if the number of tokens in place P_{UP} reaches zero.

Transitions T_{GS} and T_{LS} represent the servicing of grid and local tasks inside the resource, respectively. The service rate of each individual processor is μ which is multiplied by the number of grid (local) tasks serviced by the resource to get the firing rate of the transition. This is shown by symbol # near transitions T_{GS} and T_{LS}. After firing the timed transition T_{GS} (T_{LS}), a token is removed from place P_{GS} (P_{LS}) and deposited into place P_{UP} to show that one processor has already finished its job and can be allocated to another waiting grid/local task. The failure events of busy processors are modeled by timed transitions T_{G_BPF} and T_{L_BPF} for processors servicing grid and local tasks, respectively. The firing rates of these transitions are also marking dependent.

3.2 Second Model

In the second model, we wish to assign higher execution priority to local tasks over grid ones. According to this mechanism, a grid task can only be assigned to an idle processor if there is no local task in local queue of the resource. This modification in the model can be easily done by adding a guard function to immediate transition t_{GS} of Fig. 1 to prevent this transition to fire if there is a token in place P_{LQ}. The new guard function is described in Table 2.

3.3 Third Model

In the third model, preemptive priority scheme is considered to be applied to the simultaneous execution of grid and local tasks. In this scheme, if a newly arriving local task finds all the processors busy (there is no token in place P_{UP}), it checks the number of tokens in place P_{GS} to know whether there is a processor servicing a grid task or not. If there is, the running grid task is preempted by the new local task and the processor is allocated to that local task. The preempted grid task is returned to the grid queue to be serviced later. After servicing the local task, processor is sent back to the pool of idle processors which shows that it can be allocated to service another grid/local task. The third SRN model is shown in Fig. 2. As can be seen in this figure, in the third model, an immediate transition is added to the second model to handle preemptive priority scheme. Once a token arrives to place P_{LQ} and finds place P_{UP} empty, it checks the existence of at least one token in place P_{GS}. If there is a token in place P_{GS}, the immediate transition t_{LS2} fires and removes one token from both places P_{LQ} and P_{GS}, and deposits a token into place P_{LS} to show a newly arriving local task preempting a servicing grid task. Moreover, upon firing immediate transition t_{LS2}, a token is put in place P_{GQ} to show that the preempted grid task is returned into grid queue to be scheduled on an idle processor later. The new guard function added to the third model is defined in Table 3.

3.4 Performance Measures

Outputs of all three models are obtained by assigning appropriate reward rate to each feasible marking of SRNs, and then, computing the expected reward rates in both transient and steady state cases. Let r_i denote the reward rate assigned to marking i of the SRN models described earlier. If $\pi_i(t)$ denotes the probability for the SRN model to be in marking i at time t, then the expected reward rate at time t can be computed as $\sum_i \pi_i(t) r_i$. The expected steady state reward can be computed using the same formula by replacing $\pi_i(t)$ by π_i, representing the steady state probability for the SRN model to be in marking i. The interesting measures in the proposed models are as follows.

Mean number of waiting grid/local tasks. The mean number of waiting grid (local) tasks is given by mean number of tokens in place P_{GQ} (P_{LQ}) represented by $E[\#P_{GQ}]$ ($E[\#P_{LQ}]$). The reward rate assigned to compute $E[\#P_{GQ}]$ ($E[\#P_{LQ}]$) is $\#P_{GQ}$ ($\#P_{LQ}$) which is the number of tokens existing inside place P_{GQ} (P_{LQ}).

Blocking probability of grid task arrivals. The steady state and transient blocking probabilities of grid task arrivals, P_b and $P_b(t)$, can be computed by

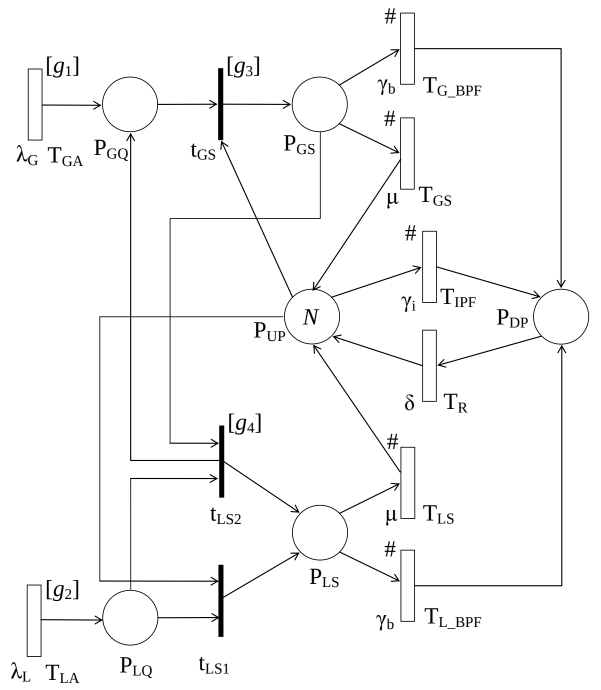

Fig. 2. Third SRN Model of a Single Grid Resource (Preemptive Priority)

assigning the following reward to SRN models.

$$r_i = \begin{cases} 1, & [\#P_{GQ}] \geq M_G \quad (1a) \\ 0, & \text{otherwise} \quad (1b) \end{cases}$$

Mean response time to grid tasks. This measure is defined as mean time from a grid task is submitted until it has been processed that is the time from the instant that a token is deposited into P_{GQ} until it is removed from P_{GS}. Using Little's law, the steady state mean response time for grid tasks, $E[R]$, can be computed as:

$$E[R] = \frac{E[\#P_{GQ}] + E[\#P_{GS}]}{\lambda_{eff}} \quad (2)$$

where λ_{eff} denotes the effective grid tasks arrival rate at grid resource and can be computed as:

$$\lambda_{eff} = (1 - p_b) \cdot \lambda_G \quad (3)$$

3.5 Numerical Results

Stochastic Petri Net Package (SPNP) [34] is used to solve the numerical examples of the proposed SRNs. All three models can be solved in a timely manner for all realistic configurations of grid resources. In the following, a sample grid resource is considered and three aforementioned measures are computed for the assumed resource. The configuration of the sample grid resource is shown in Table 4.

As can be seen in Table 4, failure rate of a running processor is four times bigger than the failure rate of an idle processor. Also, grid tasks arrival rate is assumed to be larger than local tasks arrival rate, because it is reasonable to think that a resource joins the grid environment whenever its local load is not significant [1], [2]. Since none of the open access logs reported from real grid systems contain all our required detailed information of the resources, we use random numbers to have a fair simulation. Using random numbers in this context is a common way which can be seen in [3]–[6], [8], [11], [15], [17]–[20], [26], [27], [32]. The results obtained from numerical analysis show that first SRN model results in low mean number of waiting grid tasks compared to two other models. Also, the second SRN model shows low mean number of waiting grid tasks compared to the third SRN model. Moreover, the blocking probability of grid tasks increases whenever higher execution priority is applied to local tasks.

TABLE 4
Sample Resource Configuration (Single Resource Evaluation)

Parameter	Values
Number of processors (N)	4
Failure rate of an idle processor (γ_i)	0.05
Failure rate of a busy processor (γ_b)	0.2
Repair rate of a processor (δ)	2.0
Grid tasks arrival rate (λ_G)	10.0
Local tasks arrival rate (λ_L)	8.0
Service rate of a processor (μ)	3.0
Grid queue size (M_G)	20
Local queue size (M_L)	20

Because of the lack of the space, we only present the steady state values of performance measures ignoring the transient values. The mean number of waiting grid tasks, blocking probability of grid tasks and mean response time of a single resource to grid tasks, are represented in Table 5. As can be seen in this table, all measures get their own largest values when preemptive priority scheme is applied (the third SRN model). Therefore, we can conclude that preemptive priority scheme is the worst case within three schemes considered in this paper, in the viewpoint of grid users.

3.6 Simulation Results

In order to cross-validate the results obtained from analytic models (the results reported in Subsection 3.5), we use discrete-event simulation to resolve all the models described above. A simulation is an experiment to determine characteristics of a system empirically. It is a modeling method that mimics the behavior of a system over time. The major advantage of simulation is its generality and flexibility; almost

TABLE 5
The Steady State Values of Performance Measures obtained from numerical analysis of the proposed SRN models

Measures	Steady State Values		
	Model 1	Model 2	Model 3
Mean number of waiting grid tasks	18.6566	19.0524	19.5662
Blocking probability of grid task arrivals	0.4344	0.6679	0.6683
Mean response time to grid tasks	3.6112	6.0489	6.2112

TABLE 6
The Steady State Values of Performance Measures obtained from simulating the systems corresponding to the proposed SRN models

Measures	Steady State Values		
	Model 1	Model 2	Model 3
Mean number of waiting grid tasks	18.6601	19.0904	19.5576
Blocking probability of grid task arrivals	0.4350	0.6605	0.6630
Mean response time to grid tasks	3.6137	6.0623	6.2764

any behavior of the system can be easily simulated [31], [35], [36].

The results we obtained from analytic-numeric solution of the Markov reward models underlying the SRN models and their simulative solutions are almost the same for all three measures. For the sake of brevity, only the steady state values of the performance measures gained from simulation are reported here. Table 6 shows the steady state values of the performance measures computed by averaging over at least 10 to 15 simulation runs performed for each model to generate more dependable results. The standard deviation of the results of repeated simulation runs is less than 0.01 for all models. For example, standard deviations of the results reported for $Model$ 1 in Table 6 are 0.007, 0.0007, and 0.005 for the mean number of waiting grid tasks, the blocking probability of grid tasks, and the mean response time of grid tasks, respectively.

Comparing the results reported in Table 6 and Table 5 shows that the results gained from analytic-numeric solution of the proposed SRN models are very close to the results obtained from simulating the related grid resources. For example, the relative error of the analytic-numeric and simulative solutions for $Model$ 1 which can be calculated from the first columns of Table 5 and Table 6 are 1.88×10^{-4}, 1.38×10^{-3}, and 6.92×10^{-4} for the mean number of waiting grid tasks, the blocking probability of grid tasks, and the mean response time of grid tasks, respectively.

4 SRN Models for Entire Grid Environment

As presented in Section 3, each of the grid resources can be modeled using an SRN and mean response time of the resource to grid tasks can be computed by solving the related SRN model. Although this evaluation can be useful when a single resource is considered in isolation, it is not applicable to real grid environments consisting of many single resources collaborating with each other to execute grid tasks. In other words, when several resources are working together to service a common task, the performance of each of them will be different from its performance in isolation even with the same input parameters. Actually, in this case, the effect of other resources to the resource under study should be considered to be able to accurately evaluate and predict the performance of the single resource.

In order to fulfill this requirement, we should gather all the resources to shape a cluster, and then, club the clusters together and form an entire grid. In grid environments, it is assumed that there is a RMS (or GM) which receives grid tasks from grid users, and dispatches the tasks among grid resources. Different gathering and dispatching mechanisms can be assumed for RMS. The inner structure of a RMS together with four different dispatching mechanisms for a RMS was modeled using stochastic models in [8]. Different hierarchies can be considered for RMS and grid resources. In one-level hierarchy, each of the grid resources is connected to RMS, and RMS sends grid tasks to the resources, directly. In two-level hierarchy, we can assume that there is a RMS on top of all clusters that receives grid tasks from grid users, and then dispatches tasks among clusters to be executed by the resources inside each of the clusters.

As can be seen, hierarchical topology can be easily considered in one level, and then extended to two or more levels. If we can solve the problem with one-level hierarchy, we can extend it and solve for a two-level hierarchy as well. The models presented in Section 3, can be properly adopted to model multi-level hierarchies. In the following, we show one-level hierarchy model in which grid tasks are submitted to RMS, and then dispatched among all the resources existing inside a single cluster. If this problem is solved, then each of the grid resources could be replaced with another cluster (set of homogeneous resources), and therefore, the solution would be continued to solve the problem inside that cluster. Using this way, even more levels of hierarchy can be modeled and evaluated, but generally, solving one-level hierarchy

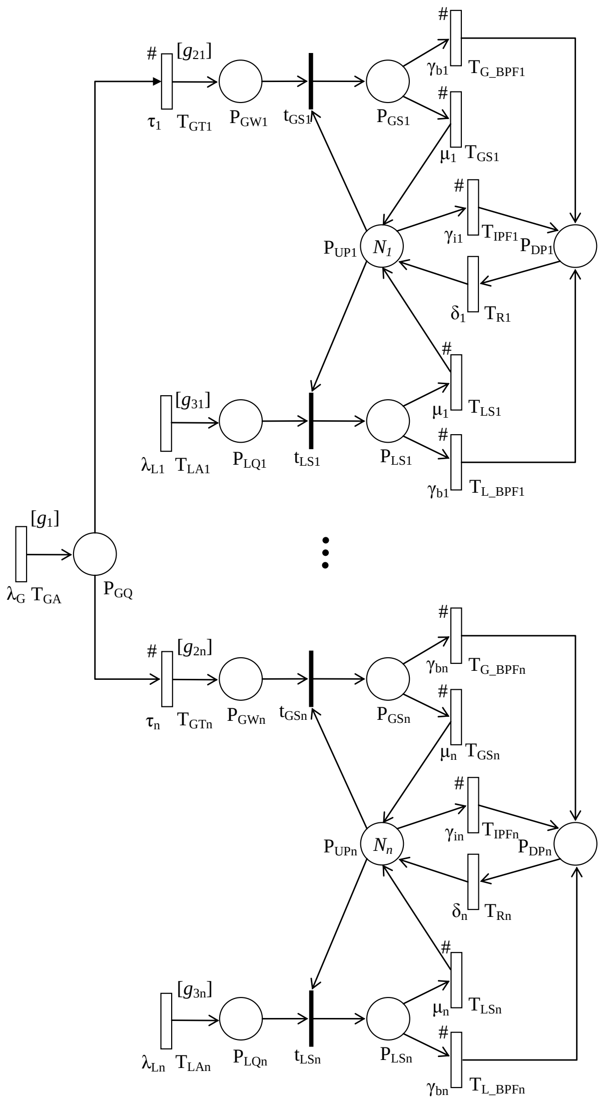

Fig. 3. Exact SRN Model of Entire Grid Environment

TABLE 7
Guard Functions of the Exact Model of Entire Grid Shown in Fig. 3

Guard Functions	Values
g_1	1 if $[\#P_{GQ}] < M_G$
	0 otherwise
$g_{2i} (1 \leq i \leq n)$	1 if $[\#P_{GWi}] = 0$
	0 otherwise
$g_{3i} (1 \leq i \leq n)$	1 if $[\#P_{LQi}] < M_{Li}$
	0 otherwise

suffices to show the correctness and applicability of the proposed model to the real grid environments.

4.1 Exact Model

Using single resource models presented in Section 3, we can shape a grid environment. To do this, one of the models should be chosen for this reason. We use the simplest model, the $First\ Model$, to explain our method for modeling and evaluating the performability of whole grid environment, but generally, each of them can be used for this reason. Figure 3 shows an SRN model of a one-level grid environment with n resources.

As can be seen in this model, there is a single entry of grid tasks for all the resources. In other words, grid queue is shared among all of the resources and each resource can execute a grid task whenever it has at least one idle processor. Place P_{GQ} in Fig. 3 represents the grid queue of the grid environment and transition T_{GA} represents the arrival process of grid tasks to the environment. The guard function g_1 takes care of the size of grid queue and does not let transition T_{GA} to fire if the number of existing tokens in place P_{GQ} reaches a given threshold, M_G. All guard functions of the model shown in Fig. 3 are described in Table 7.

Timed transition T_{GTi} is considered for each resource i to model the time needed to transfer required data of a grid task from RMS to resource i. Once a token is deposited into place P_{GQ}, existence of a token in place P_{GWi} is checked. If there is no token in place P_{GWi}, timed transition T_{GTi} is enabled. It is worthwhile to mention that if we remove place P_{GWi} and immediate transition t_{GSi} from all of the resources, and then connect place P_{UPi} to the time transition T_{GTi} in each of the resources, the model will consider higher priority to local tasks over the grid tasks inside each resource. The SRN shown in Fig. 3 can be used to model any grid environment with any number of grid resources with different configurations. However, when it is applied to model a grid environment even with a small number of resources, it encounters the largeness problem and the number of the states in underlying Markov chain becomes more than can be managed and solved by existing tools and packages. The number of the states increases drastically as the grid queue size, number of processors and local queue size of each of the resources increase. As an example, assume a grid environment with $four$ resources and each resource with only *one* processor. Also consider a single grid queue for all resources with the size equal to $five$ and a separate local queue inside each of the resources with size of *one* task. If the SRN model shown in Fig. 3 is used to solve this problem, the number of the states in the underlying Markov chain will be $781,926$. Now, if we add one more resource to this environment, the number of the states exceeds $10,000,000$! It shows that although this SRN can be used to model and evaluate the performability of a grid environment, it is impossible to use it for modeling real grids with a large number of resources. In order to cope with this difficulty, two different approximate models are presented in next two subsections.

4.2 Folded Approximate Model

Considering SRN model presented in Fig. 3, it can be concluded that the structure of all of the single resources is the same. Hence, one possible approximate model consists of 1 arbitrarily chosen tagged-resource sub-model and a sub-model for the remaining $(n-1)$ resources folded together as shown in Fig. 4. In this model, resource 1 is considered to be a tagged-resource and all the remaining resources (resource 2 to resource n) are shown with the subnet named Sub. It is assumed that all the resources are homogeneous and have the same number of processors and local queue seizes. As mentioned earlier, this assumption can be removed when the one-level hierarchy is extended to a two-level hierarchy.

Places P_{GW}, P_{GS}, P_{UP}, P_{DP}, P_{LQ} and P_{LS} correspond to places P_{GWi}, P_{GSi}, P_{UPi}, P_{DPi}, P_{LQi} and P_{LSi} $(2 \leq i \leq n)$ of exact model, respectively. The number of tokens inside place P_{UP} is $N = (n-1) \cdot N_1$ where N_1 is the number of tokens in P_{UP1} that is number of processors inside resource 1. Furthermore, grid tasks transmission rate (τ), local tasks arrival rate (λ_L), local queue size (M_L) and processor repair rate (δ) of the sub-model Sub are $\tau = (n-1) \cdot \tau_1$, $\lambda_L = (n-1) \cdot \lambda_{L1}$, $M_L = (n-1) \cdot M_{L1}$ and $\delta = (n-1) \cdot \delta_1$ where τ_1, λ_{L1}, M_{L1} and δ_1 are the corresponding parameters of resource 1. Since the failure rates of idle and busy processors (γ_i and γ_b) and service rate of the processors (μ) depend on the number of processors, we do not multiply these numbers by $(n-1)$, because the number of processors (N) has been multiplied by $(n-1)$ already. It is worthwhile to mention that multiplying the parameters of a single resource by $(n-1)$, and associating them to the corresponding parameters of the subnet Sub is only to approximate the required metrics. However, exact estimation of these parameters (especially the rates) is a difficult task since there are some interconnected factors here. The guard functions g_2 and g_3 can be written as guard functions g_{2i} and g_{3i} described in Table 7 by replacing $[\#P_{GWi}] = 0$ with $[\#P_{GW}] = 0$ and $[\#P_{LQi}] < M_{Li}$ by $[\#P_{LQ}] < M_L$.

The folded SRN shown in Fig. 4 can be used to approximate the SRN shown in Fig. 3, properly. This model encounters largeness problem much later than the exact model, but the scalability problem still exists in this model. As an example, consider the sample grid mentioned in Subsection 4.1. If we increase the number of the resources to 10, the number of the states in the underlying Markov chain of SRN model shown in Fig. 4 will be $5,019,235$. If we add only one resource to the environment, the number of the states will increase to $7,380,516$. As can be seen, this model still is not appropriate for very large-scale grids with large number of resources; however it provides a good improvement over the exact model regarding to the number of states in the underlying Markov chain. It should be noted that the exact model cannot handle a system with more than 4 or 5 resources, but the folded model can handle a small system with about 15-20 resources (like a cluster). But, the scalability problem would be much severe if the number of the processors inside each of the resources and grid (local) queue size of the environment (resource) increase.

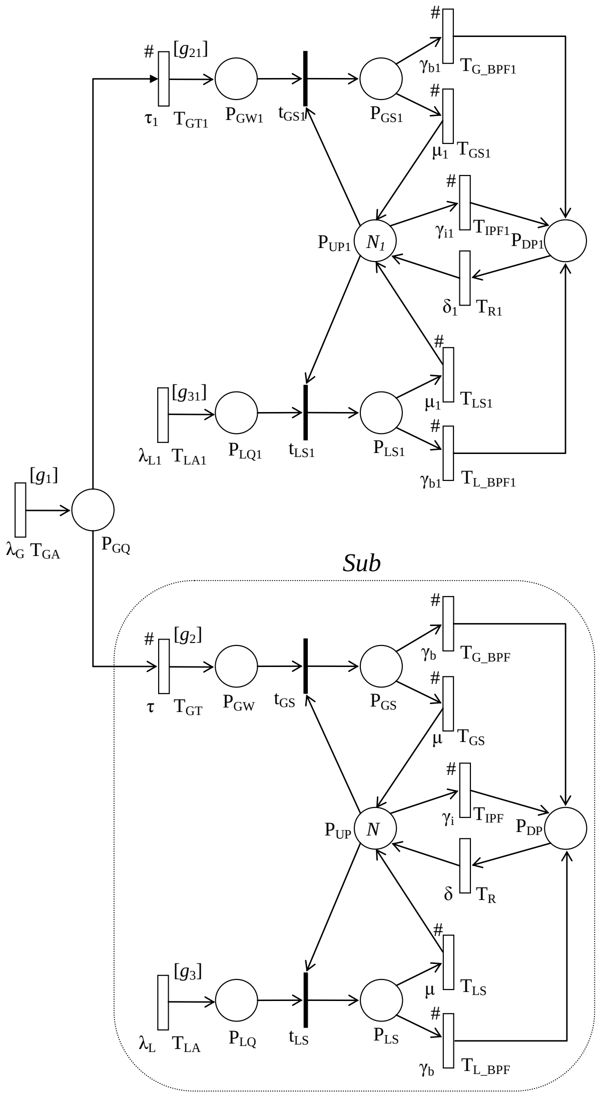

Fig. 4. Folded SRN Model of Entire Grid Environment

4.3 Fixed-point Approximate Model

In order to deal with the largeness problem of the folded model and to refine the model one step further to be able to solve large-scale grids, a better approximate model based on fixed-point iteration [17]–[20], [24], [25] is presented. The fixed-point iteration method is known as a good solution for analyzing a system with some interrelated subsystems. Each subsystem is analyzed with the remainder of the system represented in a simplified manner. The method acts iteratively and the parameters of the simplified complement of a subsystem are modified after the other subsystems are analyzed. The subsystem is then re-analyzed with new input parameters to produce new inputs for other subsystems. This procedure is repeated till the difference between two successive iterations is below a certain tolerance level.

We divide the whole model into two groups; a tagged-resource together with grid arrivals (RMS) sub-model and a sub-model for remaining $(n-1)$ resources. Since the remaining $(n-1)$ resources act as delay with respect to the other sub-model, we can replace it with a single timed transition in the overall model. Figure 5 shows one tagged-resource together with RMS in which the $(n-1)$ remaining resources are approximated with a single timed transition named T_D. All the rates, probabilities, the number of tokens and guard functions existing in SRN shown in Fig. 5 are the same as their corresponding values in the model of Fig. 4. To be able to evaluate mean response time of a grid resource to grid tasks, we should solve this model with an appropriate firing rate for transition T_D (α). Since we do not know how long this delay is, we cannot solve the SRN model shown in Fig. 5 directly. Instead, we solve it using an iterative method by exploiting the SRN presented in Fig. 6 modeling $(n-1)$ remaining resources.

In Fig. 6, place P_{GQ} contains M tokens where $1 \leq M \leq M_G$ and M_G is the grid queue size. The grid tasks transmission rate (τ), local tasks arrival rate (λ_L), local queue size (M_L) and processor repair rate (δ) of SRN model represented in Fig. 6 are $\tau = (n-1)\cdot\tau_1$, $\lambda_L = (n-1)\cdot\lambda_{L1}$, $M_L = (n-1)\cdot M_{L1}$ and $\delta = (n-1)\cdot\delta_1$ where τ_1, λ_{L1}, M_{L1} and δ_1 are the corresponding parameters of the tagged-resource existing in SRN of Fig. 5.

As can be seen in Fig. 6, a new place P_T is added to the SRN to trap all the grid tasks submitted to the system after their successful execution or failure of the processor assigned to execute them. Therefore, if there are M tokens in place P_{GQ}, after a specific amount of time, all these tokens will be moved to place P_T. If we define suitable guard functions for all the transitions of this model, we can make the underlying Markov chain as an absorbing Markov chain. The guard functions of SRN model of Fig. 6 are described in Table 8.

Applying the guard functions described in Table 8, we can make sure that the underlying Markov chain of SRN model presented in Fig. 6 is an absorbing Markov chain. Therefore, we can compute the mean time to absorption of this SRN. Define $MTTA_i$ as mean time to absorption of SRN shown in Fig. 6 when M is equal to i. Also, let $\pi(P_{GQ} = i)$ denote the steady state probability of there being i tokens in place P_{GQ} of SRN model of Fig. 6 which is computed using the steady state analysis of SRN model of Fig. 5. Hence, the mean time to absorption of the SRN shown in Fig. 6 $(MTTA)$ can be computed as:

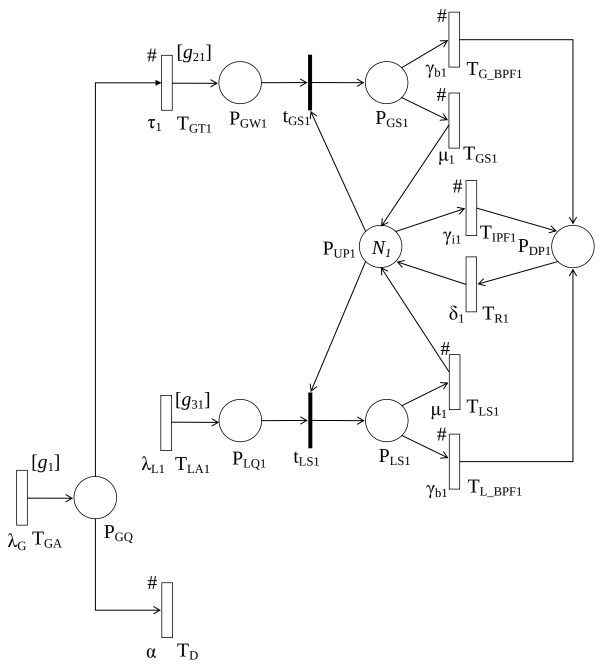

Fig. 5. Fixed-point Approximate Model for Entire Grid (sub-model 1)

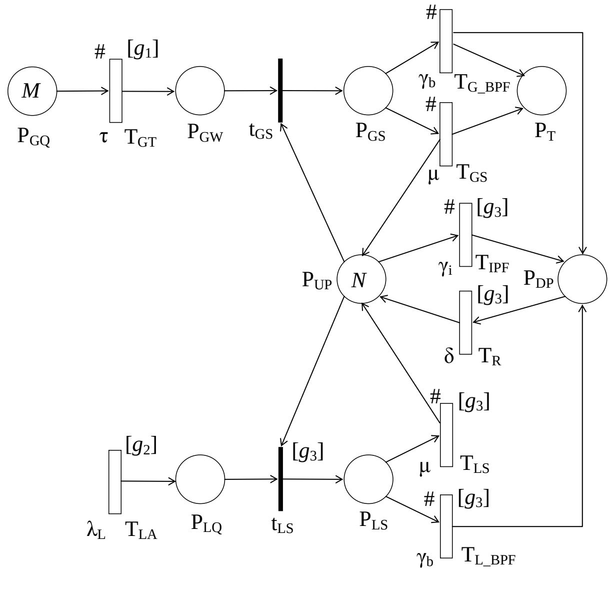

Fig. 6. Fixed-point Approximate Model for Entire Grid (sub-model 2)

TABLE 8
Guard Functions of SRN Model Shown in Fig. 6

Guard Functions	Values
g_1	1 if $[\#P_{GW}] = 0$
	0 otherwise
g_2	1 if $[\#P_{LQ}] < M_L$ and $[\#P_T] < M$
	0 otherwise
g_3	1 if $[\#P_T] < M$
	0 otherwise

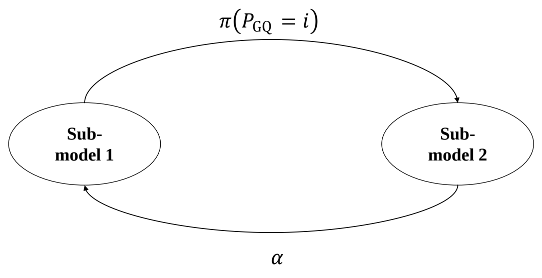

Fig. 7. Interactions Between the Sub-models of Fixed-point Approximate SRN Model as an Import Graph

$$MTTA = \sum_{i=1}^{M_G} MTTA_i \cdot \pi(P_{GQ} = i) \tag{4}$$

Once we compute $MTTA$, we can compute firing rate of timed transition T_D as (5) which represents the firing rate due to all resources other than tagged-resource.

$$\alpha = \frac{1}{MTTA} \tag{5}$$

This value (α) is used when we solve the SRN model shown in Fig. 5 at the next iteration. After solving SRN model of Fig. 5, the new steady state probability of there being i tokens in place P_{GQ} is computed, and then used in (4) to calculate the new value of α. This procedure continues till the difference between two successive values of α reaches a given threshold. Two sub-models are represented in Fig. 5 and Fig. 6 and their interactions are shown as an import graph in Fig. 7.

4.4 Numerical Results

In order to compare the results obtained from three SRN models proposed for an entire grid, a sample grid environment with different number of resources is considered in this subsection. Configuration parameters of the system are shown in Table 9. The values shown in Table 9 are random numbers which can be easily replaced with the corresponding numbers of real systems. They are used only to show how the SRN models work. As mentioned in Subsection 3.5, this is a common way to show the applicability of the models to real systems. We apply the exact model (Fig. 3), folded approximate model (Fig. 4) and fixed-point approximate model (Fig. 5 and Fig. 6) to evaluate the sample grid environment.

The comparison of the steady state mean response time of grid tasks between exact model and two other approximate models, folded and fixed-point, are presented in Table 10. Moreover, the steady state blocking probability resulted from the exact and two approximate models are presented in Table 11. As can be seen in Table 10, the steady state mean response time of grid tasks obtained from all models are very close to each other. Also, the differences between the results reported in Table 11 show that our proposed approximate models (especially fixed-point model) can appropriately estimate the steady state blocking probability of the exact model. It should be mentioned that running the exact model for only 5 grid resources produces more than $10,000,000$ states and cannot be solved in a timely manner. Therefore, we only compare the performance measures of the models with 2 to 4 numbers of resources.

Similar to the SRN models proposed for a single grid resource, and in order to validate the results obtained from exact model and compare it with the results obtained from approximate models, we simulate the grid environment mentioned above. Because of the lack of the space, we do not present the simulation results here, but as a short report, it should be mentioned that the relative errors between the results of exact model and the average simulation runs for blocking probability of grid tasks are 1.48×10^{-4}, 1.77×10^{-4}, and 4.54×10^{-5} for $N = 2$, 3, and 4, respectively. In addition to validate the results of the exact model, one of the most important advantages of simulation is being able to simulate a grid environment with a more number of grid resources and comparing the results with the approximate models which show the proposed folded and fixed-point models appropriately approximate a real environment.

The numbers of the states and non-zero entries of the Markov chain matrix generated by three exact, folded and fixed-point approximate SRN models are shown in Table 12 and Table 13, respectively. The numbers reported for fixed-point approximate model are only the numbers generated by SRN model shown in Fig. 6, because the numbers of the states and non-zero entries of the underlying Markov chain of SRN model of Fig. 5 are fixed numbers for all settings. As can be seen in Table 12 and Table 13, numbers

TABLE 9
Sample Resource Configuration (Entire Grid Evaluation)

Parameter	Values
Number of processors in each resource (N_1)	1
Failure rate of an idle processor (γ_{i1})	0.05
Failure rate of a busy processor (γ_{b1})	0.1
Repair rate of a processor (δ_1)	2.0
Grid tasks arrival rate (λ_G)	20.0
Grid tasks transmission rate (τ_1)	4.0
Local tasks arrival rate (λ_{L1})	7.0
Service rate of a processor (μ_1)	3.0
Grid queue size (M_G)	5
Local queue size (M_{L1})	2

TABLE 10
The Steady State Mean Response Times From the Exact and Approximate Models

Number of resources	SRN Models		
	Exact	Folded	Fixed-point
2	2.176898651	2.176898651	2.173223694
3	1.430419422	1.377265141	1.419331918
4	1.054943360	0.996678592	1.084381780

TABLE 11
The Steady State Blocking Probability of Grid Tasks From the Exact and Approximate Models

Number of resources	SRN Models		
	Exact	Folded	Fixed-point
2	0.856938653	0.856938653	0.856724326
3	0.785753252	0.778123897	0.784162732
4	0.714924782	0.700370246	0.721953578

reported for fixed-point approximate model are much less than those for two other models. It shows that we can use this SRN (and sometimes folded SRN) to model real grid environments.

In the experiments mentioned above, the initial value for firing rate of timed transition T_D (α) is set to 0.2. After some iterations, it converges to the suitable value in each experiment based on the input parameters. Figure 8 shows the convergence of this parameter to the final value based on the number of iterations only for 5 successive iterations. As can be seen in Fig. 8, the proposed fixed-point approximate model converges to the final value very fast. In all our experiments with SRN models given in Fig. 5 and Fig. 6, we have seen very good convergence behavior as shown in Fig. 8.

TABLE 12
Number of the States in the Underlying Markov Chain of the Exact, Folded and Fixed-point Approximate Models

Number of resources	SRN Models		
	Exact	Folded	Fixed-point
2	2,166	2,166	92
3	41,154	10,716	295
4	781,926	33,060	651

TABLE 13
Number of the Non-zero Entries in the Underlying Markov Chain Matrix of the Exact, Folded and Fixed-point Approximate Models

Number of resources	SRN Models		
	Exact	Folded	Fixed-point
2	15,105	15,105	287
3	413,345	91,458	1,275
4	10,254,205	314,566	3,268

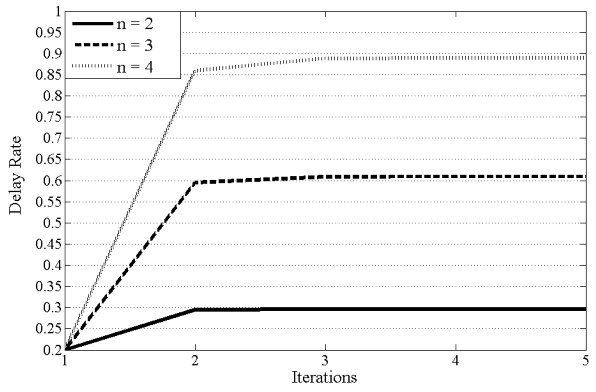

Fig. 8. Convergence of Delay Rate of Timed Transition T_D to Its Final Value in Fixed-point Approximate Model

4.5 Existence of a Fixed-point

In theory, we should provide a proof to show that a fixed-point always exists when a decomposition of this form is made. To do this, the following proof is presented.

Define $y_i = \pi(P_{GQ} = i)$ where $\pi(P_{GQ} = i)$ is the steady state probability of there being i tokens in place P_{GQ} and $1 \leq i \leq M_G$. Considering the SRN model shown in Fig. 5 and its underlying Markov chain, it is obvious that for some function h we can write $y_i = h(\alpha)$ where α is the firing rate of timed transition T_D. On the other hand, this rate is obtained by computing mean time to absorption in SRN shown in Fig. 6 for i number of tokens inside place P_{GQ} where $1 \leq i \leq M_G$. As discussed earlier, using (4) and (5), we can write $\frac{1}{\alpha} = \sum_{i=1}^{M_G} MTTA_i \cdot \pi(P_{GQ} = i)$ where $MTTA_i$ is mean time to absorption of SRN shown in Fig. 6 when number of the tokens in place P_{GQ} is equal to i. Let the vector $\vec{y} = (y_1, \ldots, y_{M_G})$, so for some function f we can write $\vec{y} = f(\vec{y})$.

Now, Brouwers fixed-point theorem [24] is used to show that $f(\vec{y}) = \vec{y}$ has a solution. This theorem states that if there exists a compact, convex set $\overline{C} \subset R^n$ and there exists a continuous function f such that $f(\vec{y}) \in \overline{C}$ for all $\vec{y} \in \overline{C}$ then there exist a solution to equation $f(\vec{y}) = \vec{y}$.

Since $\vec{y} = (y_1, y_2, \ldots, y_{M_G})$ and each $y_i = \pi(P_{GQ} = i)$ representing the steady state probability of there being i tokens in place P_{GQ} is bounded below by 0 and above by 1, each y_i is bounded in range $[0, 1]$. Hence, $\overline{C}$ is defined to be the set of points $(y_1, y_2, \ldots, y_{M_G})$ where each $y_i \in [0, 1]$. Consider the function f over $\overline{C}$ by defining $y_i = \pi(P_{GQ} = i)$ and $y_i = f(y_i)$, $1 \leq i \leq M_G$. Since the probability of being in a subset of a Markov chain is always bounded below by 0 and above by 1, therefore $y_i \in [0, 1]$ for all i.

Now $\overline{C}$ will be shown to be a convex set. A set $\overline{C} \subset R^n$ is convex if $\lambda\vec{x} + (1-\lambda)\vec{y} \in \overline{C}$ whenever $\vec{x}$ and $\vec{y}$ are n-vectors $\in \overline{C}$ and $\lambda \in [0,1]$. Let's consider one equation: $z_i = \lambda x_i + (1-\lambda)y_i$. $z_i \geq 0$ since $\lambda \geq 0$, $x_i \geq 0$, $(1-\lambda) \geq 0$, and $y_i \geq 0$. Since z_i is maximized when $x_i = y_i = 1$ (since $\lambda \geq 0$ and $(1-\lambda) \geq 0$) to its maximum value 1, $z_i \leq 1$.

Finally, we will show that f is continuous over $\overline{C}$. Function $f(\vec{y})$ is continuous if for each point $\hat{y} \in \overline{C}$, $\lim_{\vec{y}\to\hat{y}} f(\vec{y}) = f(\hat{y})$. As $f(\vec{y})$ is a vector valued function, this is equivalent to saying $\lim_{\vec{y}\to\hat{y}} f_k(\vec{y}) = f_k(\hat{y})$ for $k \in \{1,2,\ldots,M_G\}$ and $\hat{y} \in \overline{C}$. By defining $\lim_{\vec{y}\to\hat{y}} f_k(\vec{y}) = \lim_{\vec{y}\to\hat{y}} \big[[MTTA_1 \cdot \pi(P_{GQ} = 1)] + \ldots + [MTTA_{M_G} \cdot \pi(P_{GQ} = M_G)]\big]$ we can derive $\lim_{\vec{y}\to\hat{y}} f_k(\vec{y}) = \lim_{\vec{y}\to\hat{y}} \big[[MTTA_1 \cdot y_1] + \ldots + [MTTA_{M_G} \cdot y_{M_G}]\big]$. Since $y_i = \pi(P_{GQ} = i)$ is the probability of being in a subset of the states of a Markov chain, $\lim_{y_i\to\hat{y}_i} y_i = \hat{y}_i$. Since each term of the summation converges to its (finite) value at $\hat{y}$, $\lim_{\vec{y}\to\hat{y}} f_k(\vec{y}) = f_k(\hat{y})$ and therefore $\lim_{\vec{y}\to\hat{y}} f(\vec{y}) = f(\hat{y})$.

5 Conclusion and Future Work

Isolated performance and availability evaluation of a grid resource may cause undependable results since these two factors influence each other in many cases. Therefore, to reach a more realistic analysis and more dependable results, the performance and availability of a grid resource should be evaluated simultaneously. To do this, SRNs are used in this paper to model and analyze the composite performance and availability of a single grid resource. The proposed SRNs consider three different scheduling schemes to simultaneously schedule grid and local tasks among the processors existing in a single resource. After successful evaluation of performability of a single grid resource, the SRN models are combined to capture an entire grid environment. Since the general SRN model encounters the largeness problem, two approximate models are proposed to solve this problem. Using the approximate models, large scale grid environments can be modeled and evaluated properly. Some illustrative examples are given to show the application of the proposed SRNs to actual systems. Therefore, the proposed models can be appropriately used in the analysis and design phases of real grid systems to evaluate the performability of the systems before implementation and development. Furthermore, the models can be used to evaluate the performability of currently running grids and study the behavior of the system when some changes are made.

There is a number of research issues remaining open for future work. One of the most interesting issues is using the single resource performability models to design a scheduling algorithm to dispatch grid tasks among the resources. Using these models, we can compute the probability and its corresponding effective service rate of a single resource to grid tasks. Having these measures for all the resources existing in the grid environment, we can use heuristic algorithms to dispatch grid tasks to the resources to optimize the general performability of a grid service. To do this, a new measure for general performability should be defined. One possible measure would be used here is the expected service time given that the service does not fail. This is what we are working on currently. As another open problem in this field, one can use the SRN models to find response time distribution in a grid environment. In this paper, we compute mean response time of a grid resource to grid tasks, but finding the distribution of response time is more useful. Applying more sophisticated scheduling algorithms between grid and local tasks is another open problem in this area.

Acknowledgment

The authors would like to thank Dr. Xiaoyan Yin, Ms. Maryam Bagheri and Ms. Saeedeh Mehri for their support and time given in working with us.

References

[1] I. Foster and C. Kesselman, *The Grid 2: Blueprint for a New Computing Infrastructure*, 2nd ed. Morgan Kaufmann, 2004.

[2] K. Krauter, R. Buyya, and M. Maheswaran, "A taxonomy and survey of grid resource management systems for distributed computing," *Software: Practice and Experience*, vol. 32, no. 2, pp. 135–164, 2002.

[3] Y.-S. Dai and G. Levitin, "Reliability and performance of tree-structured grid services," *IEEE Transactions on Reliability*, vol. 55, no. 2, pp. 337–349, 2006.

[4] G. Levitin and Y.-S. Dai, "Service reliability and performance in grid system with star topology," *Reliability Engineering and System Safety*, vol. 92, no. 1, pp. 40–46, 2007.

[5] Y.-S. Dai and G. Levitin, "Optimal resource allocation for maximizing performance and reliability in tree-structured grid services," *IEEE Transactions on Reliability*, vol. 56, no. 3, pp. 444–453, 2007.

[6] M. A. Azgomi and R. Entezari-Maleki, "Task scheduling modelling and reliability evaluation of grid services using coloured Petri nets," *Future Generation Computer Systems*, vol. 26, no. 8, pp. 1141–1150, 2010.

[7] E. Caron, V. Garonne, and A. Tsaregorodtsev, "Definition, modeling and simulation of a grid computing scheduling system for high throughput computing," *Future Generation Computer Systems*, vol. 23, no. 8, pp. 968–976, 2007.

[8] R. Entezari-Maleki and A. Movaghar, "Availability modeling of grid computing environments using SANs," in *The 19th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2011)*, Split, Croatia, September 2011, pp. 1–6.

[9] R.-S. Chang, C.-Y. Lin, and C.-F. Lin, "An adaptive scoring job scheduling algorithm for grid computing," *Information Sciences*, vol. 207, no. 1, pp. 79–89, 2012.

[10] R. Entezari-Maleki and A. Movaghar, "A probabilistic task scheduling method for grid environments," *Future Generation Computer Systems*, vol. 28, no. 3, pp. 513–524, 2012.

[11] S. Parsa and R. Entezari-Maleki, "Task dispatching approach to reduce the number of waiting tasks in grid environments," *The Journal of Supercomputing*, vol. 59, no. 1, pp. 469–485, 2012.

[12] J. F. Meyer, "On evaluating the performability of degradable computing systems," *IEEE Transactions on Computers*, vol. C-29, no. 8, pp. 720–731, 1980.

[13] R. M. Smith, K. S. Trivedi, and A. V. Ramesh, "Performability analysis: measures, an algorithm and a case study," *IEEE Transactions on Computers*, vol. 37, no. 4, pp. 406–417, 1988.

[14] D. M. Nicol, , W. H. Sanders, and K. S. Trivedi, "Model-based evaluation: From dependability to security," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 48–65, 2004.

[15] O. C. Ibe, H. Choi, and K. S. Trivedi, "Performance evaluation of client-server systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 4, no. 11, pp. 1217–1229, 1993.

[16] G. Ciardo and K. S. Trivedi, "A decomposition approach for stochastic reward net models," *Performance Evaluation*, vol. 18, no. 1, pp. 37–59, 1993.

[17] H. Choi and K. S. Trivedi, "Approximate performance models of polling systems using stochastic Petri nets," in *The Eleventh Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '92)*, Florence, Italy, May 1992, pp. 2306–2314.

[18] L. A. Tomek and K. S. Trivedi, "Fixed point iteration in availability modeling." in *Fault-Tolerant Computing Systems*, ser. Informatik-Fachberichte, M. D. Cin and W. Hohl, Eds., vol. 283. Springer, 1991, pp. 229–240.

[19] Y. Ma, J. J. Han, and K. S. Trivedi, "Channel allocation with recovery strategy in wireless networks," *European Transactions on Telecommunications*, vol. 11, no. 4, pp. 395–406, 2000.

[20] V. Mainkar and K. S. Trivedi, "Approximate analysis of priority scheduling systems using stochastic reward nets," in *The 13th International Conference on Distributed Computing Systems (ICDCS '93)*, Pittsburgh, Pennsylvania, USA, May 1993, pp. 466–473.

[21] H. Sun and K. S. Trivedi, "A stochastic reward net model for performance analysis of prioritized DQDB MAN," *Computer Communications*, vol. 22, no. 9, pp. 858–870, 1999.

[22] F. Longo, R. Ghosh, V. K. Naik, and K. S. Trivedi, "A scalable availability model for infrastructure-as-a-service cloud," in *The 41st IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2011)*, Hong Kong, June 2011, pp. 335–346.

[23] R. Ghosh, F. Longo, V. K. Naik, and K. S. Trivedi, "Modeling and performance analysis of large scale IaaS clouds," *Future Generation Computer Systems*, In Press, DOI: http://dx.doi.org/10.1016/j.future.2012.06.005, 2012.

[24] V. Mainkar and K. S. Trivedi, "Sufficient conditions for existence of a fixed point in stochastic reward net-based iterative models," *IEEE Transactions on Software Engineering*, vol. 22, no. 9, pp. 640–653, 1996.

[25] J. K. Muppala and K. S. Trivedi, "Composite performance and availability analysis using a hierarchy of stochastic reward nets," in *Computer Performance Evaluation, Modelling Techniques and Tools*, G. Balbo and G. Serazzi, Eds. Elsevier Science Publishers B.V. (North-Holland), 1992, pp. 335–349.

[26] S. Parsa and R. Entezari-Maleki, "A queuing network model for minimizing the total makespan of computational grids," *Computers and Electrical Engineering*, vol. 38, no. 4, pp. 827–839, 2012.

[27] O. C. Ibe and K. S. Trivedi, "Stochastic Petri net models of polling systems," *IEEE Journal on Selected Areas in Communications*, vol. 8, no. 9, pp. 1649–1657, 1990.

[28] J. L. Peterson, *Petri Net Theory and the Modeling of Systems*, 1st ed. Prentice Hall, 1981.

[29] F. Bause and P. S. Kritzinger, *Stochastic Petri Nets: An Introduction to the Theory*, 2nd ed. Vieweg+Teubner Verlag, 2002.

[30] M. Ajmone Marsan, G. Conte, and G. Balbo, "A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems," *ACM Transactions on Computer Systems*, vol. 2, no. 2, pp. 93–122, 1984.

[31] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis, *Modeling with Generalized Stochastic Petri Nets*, 1st ed. John Wiley and Sons, 1995.

[32] J. K. Muppala, K. S. Trivedi, V. Mainkar, and V. G. Kulkarni, "Numerical computation of response time distributions using stochastic reward nets," *Annals of Operations Research*, vol. 48, no. 2, pp. 155–184, 1994.

[33] S. Jafar, A. Krings, and T. Gautier, "Flexible rollback recovery in dynamic heterogeneous grid computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 6, no. 1, pp. 32–44, 2009.

[34] G. Ciardo, J. K. Muppala, and K. S. Trivedi, "SPNP: stochastic Petri net package," in *The 3rd International Workshop on Petri Nets and Performance Models (PNPM '89)*, Kyoto, Japan, December 1989, pp. 142–151.

[35] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*, 2nd ed. John Wiley & Sons, 2006.

[36] K. Kant and M. M. Srinivasan, *Introduction to Computer System Performance Evaluation*, 1st ed. McGraw-Hill, 1992.

Reza Entezari-Maleki is currently a Ph.D. student in Computer Engineering (Software discipline) at the Department of Computer Engineering in Sharif University of Technology, Tehran, Iran. He received his B.S. and M.S. degrees in Computer Engineering (Software discipline) from Iran University of Science and Technology (IUST), Tehran, Iran in 2007 and 2009, respectively. He is also a member of Iranian National Elite Foundation. His main research interests are grid computing, performance evaluation, performability and dependability modeling, and task scheduling algorithms.

Ali Movaghar is a Professor in the Department of Computer Engineering at Sharif University of Technology in Tehran, Iran and has been on the Sharif faculty since 1993. He received his B.S. degree in Electrical Engineering from the University of Tehran in 1977, and M.S. and Ph.D. degrees in Computer, Information, and Control Engineering from the University of Michigan, Ann Arbor, in 1979 and 1985, respectively. He visited the Institut National de Recherche en Informatique et en Automatique in Paris, France and the Department of Electrical Engineering and Computer Science at the University of California, Irvine in 1984 and 2011, respectively, worked at AT&T Information Systems in Naperville, IL in 1985-1986, and taught at the University of Michigan, Ann Arbor in 1987-1989. His research interests include performance/dependability modeling and formal verification of wireless networks and distributed real-time systems. He is a senior member of the IEEE and the ACM.

Kishor S. Trivedi received the M.S. and Ph.D. degrees in computer science from the University of Illinois, Urbana. He holds the Hudson Chair with the Department of Electrical and Computer Engineering, Duke University, Durham, NC. Since 1975, he has been with the Duke faculty. He is the author of the well-known text entitled Probability and Statistics with Reliability, Queuing and Computer Science Applications (Prentice-Hall); a thoroughly revised second edition (including its Indian edition) of this book has been published by John Wiley. He is also the author of two other books, one entitled Performance and Reliability Analysis of Computer Systems (Kluwer) and the other entitled Queuing Networks and Markov Chains (John Wiley). He has published more than 500 papers and has supervised 45 Ph.D. dissertations. He works closely with industry in carrying reliability/availability analysis and providing short courses on reliability, availability, performability modeling, and development and dissemination of software packages such as SHARPE and SPNP. His research interests are reliability, availability, performance, performability, and survivability modeling of computer and communication systems. Dr. Trivedi is a Fellow of the IEEE and is a Golden Core Member of the IEEE Computer Society. He was the recipient of the IEEE Computer Society Technical Achievement Award for his research on software aging and rejuvenation.