Addressing the Robustness of Resource Allocation in the Presence of Application and System Irregularities via PEPA Based Modeling

Srishti Srivastava[1], Ioana Banicescu[2], and William S. Sanders[2]

[1] University of Southern Indiana, Evansville IN 47712, USA
fsrishti@usi.edu
[2] Mississippi State University, Mississippi State MS 39762, USA
{ioana@cse.,wss2}@msstate.edu

Abstract. Applications executing in heterogeneous parallel and/or distributed computing (PDC) environments are often prone to unpredictable runtime due to variations in problem, algorithm, and system characteristics. This serves as a key motivation towards a study of the robustness of resource allocations required to maintain and guarantee a desired level of performance. Performance modeling and evaluation is often utilized to understand and predict the behavior of the application and the computational system from a performance point of view. In prior work, performance modeling for evaluating response times of *static* resource allocations in PDC systems was introduced by the authors as a proof of concept for validating the use of the performance evaluation process algebra (PEPA) for analyzing the robustness of static resource allocations. Herein, the authors present numerical modeling of several *static* resource allocations to evaluate their robustness in the presence of *compound perturbations* generated as combinations of variations in *application workload* and *machine availability*. The novelty of the approach is to introduce the *compound effect* as the variability of both, *application workload* and *processor/machine availability*, into the performance modeling of the overall computational system. The performance is obtained as a parallel execution time via a numerical analysis of the modeled execution of applications on *non-dedicated* parallel computational resources. A significant improvement in the robustness value (up to 143%) among the mappings yielding equal parallel execution times has been demonstrated via the analysis of the results. This notable difference in the robustness values strongly indicates the benefit of selecting one mapping versus the other for guaranteeing the best execution performance.

Keywords: performance modeling and evaluation · robustness analysis · process algebra.

1 Introduction

Often in parallel and distributed computing (PDC) environments, the applications are expected to undergo variations in workload, and the underlying computational resource is considered to be non-dedicated at runtime. Therefore,

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

appropriate initial resource allocation algorithms are required for an efficient mapping of applications to machines. In addition to the traditional performance metrics (execution time, speedup, efficiency, and others), there is a need for the study of a metric of the robustness of resource allocations to guarantee a desired level of performance. Performance modeling and evaluation is often utilized to understand the behavior of concurrent and parallel computing and communication systems by identifying features of the system that are sensitive from a performance point of view. When compared to direct experiments and simulations, numerical models and the corresponding analyses are easier to reproduce, do not incur any setup or installation costs, do not impose any prerequisites for learning a simulation framework, and are not limited by the complexity of the underlying infrastructure or simulation libraries. To the best of our knowledge, performance modeling for evaluating *response times* of *static* resource allocations in parallel and distributed computing systems and the related robustness analysis remained an open problem until the authors introduced the first solution by utilizing the performance evaluation process algebra (PEPA) [5] for analyzing the robustness of static resource allocations [22][23][24].

The main contribution of this work is to study a number of *static* resource allocations via a PEPA based numerical analysis of performance modeling of the parallel execution of applications with *varying workload* on *non-dedicated* parallel computing resources with *varying machine availability*. The robustness of the resource allocations is evaluated against the *compound effect*, which is defined as the combination of the impacts of the variations in the application workload and those of machine availability on system performance. Note that in this study, the terms *processor availability* and *machine availability* are used interchangeably throughout the paper. Prior validation of our PEPA models, and a confirmation of the similarity in results of our numerical analysis with the experimental results of earlier reported research results available in the existing literature [1] are illustrated in Figure 1. However, in our prior validation work [23] only the variability in problem characteristics has been considered to mimic the experiments in [1].

In this work, the robustness value for each resource allocation is calculated as the minimum probability of the machines to finish before a desired makespan goal. Based on the analysis of the results, a number of mappings yield equal execution performance. However, a significant improvement in the robustness value (up to 143%) among the mappings yielding equal parallel execution times has been observed. Thus, this notable 2.43 times increase of robustness strongly indicates the benefits of selecting one mapping versus the other for *guaranteeing* the best execution performance. The work with PEPA can also be extended to modeling a single utility function that includes metrics, such as robustness, power consumption, and others. In general, this work is applicable to various types of computing and communication environments.

The rest of the paper is organized as follows. A description of the robustness of resource allocations in parallel and distributed computing systems and of the performance modeling techniques for numerical analysis of performance in par-

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

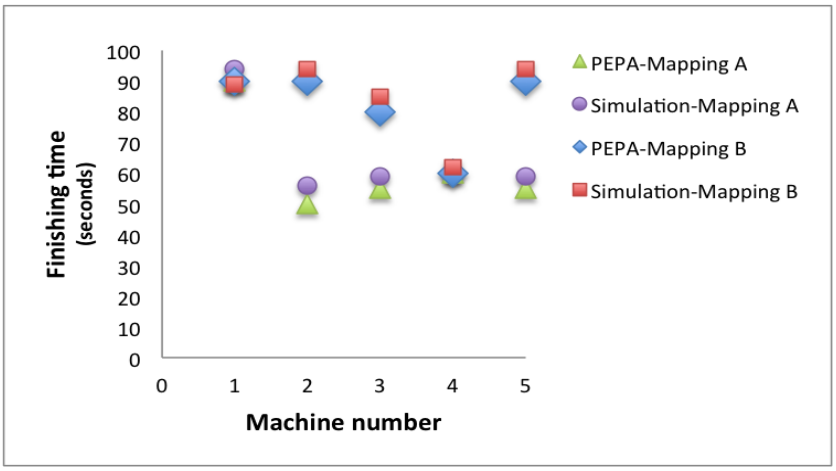


Fig. 1: A comparative analysis of the numerical results of performance modeling with existing simulation results used for validation of approach in [23].

allel computing systems are given in Section 2. The design and the use of PEPA for the numerical evaluation of the analytical models of resource allocations in parallel computing systems are presented in Section 3. A comparison of the numerical results of robustness of a number of resource allocations obtained via analytical modeling using the PEPA tool, is discussed in Section 4. Conclusions and possible future work are summarized in Section 5.

2 Background and Related Work

In general, the *mapping problem*, defined as finding the best allocation of independent tasks (or applications) onto a set of parallel processors, is known to be NP-Complete [6][7][8]. A number of research efforts have been made towards achieving robust mapping, or resource allocation techniques in parallel and distributed computing systems. Key work done in this area is being reviewed and is presented in this section. In addition, a survey of the work in performance evaluation of computer and communication systems using analytical and numerical modeling is also being discussed.

2.1 Robustness of Resource Allocation and Application Scheduling

The initial work on robust scheduling originated from job-shop application scheduling frameworks [9]. A Standard branch and bound approach was used to solve the NP-Hard robust scheduling problem (RSP) to obtain a robust schedule for N independent jobs on a single machine [10]. A number of approaches have been developed to obtain an initial robust resource allocation by utilizing optimization techniques such as, stochastic mixed integer programming [11], iterative integer programming [12], and others. In addition to designing robust

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

resource allocations, robustness metrics have also been formulated to study the performance guarantee of available static resource allocations against possible inadequate or variable computational environmental factors. A general methodology, called the Feature Perturbation Impact Analysis (FePIA) procedure, for developing robustness metrics for resource allocation has been presented by Ali et al. [1]. The authors define a resource allocation to be robust with respect to specific system *performance features* against *perturbations* (uncertainties) in specified *system parameters* if degradation in these features is constrained when limited perturbations occur [1][2]. To address this issue of investigating the robustness of scheduling techniques at the application level, together with studies conducted at the system level for a holistic approach of robustness, research has also been conducted towards analyzing the robustness of a number of dynamic loop scheduling (DLS) algorithms, which are effective in dynamic scheduling of scientific applications on large-scale parallel and distributed systems, in the presence of varying processor loads (where robustness is quantified using the *flexibility metric*) and processor failures (where robustness is quantified using the *resilience metric*) [3][4].

2.2 Process Algebra for Performance Evaluation

Analytical and numerical modeling for performance evaluation allows derivation of an expression of the performance feature of interest in terms of the input parameters of the model. In case of a predictive analysis of a computing system, analytical models generally provide the best insight into the effects of various perturbation parameters on system performance, and are easier to replicate for a comparative analysis of different systems. Markovian models have been shown to be an effective tool for performance analysis of computer and communication systems, where the system components are modeled as Markov processes, and the overall performance (for example, throughput, resource utilization, and others) is evaluated upon the numerical analysis of these Markov processes [15]. Process algebras are abstract languages used for specification and design of such systems.

PEPA, a stochastic process algebra, has been used for performance modeling and analysis of a wide range of concurrent systems. In a recent research study related to the scheduling of pipeline applications on grid resources, the PEPA workbench [16] has been used to solve the performance models of a scheduling system to obtain relevant performance information required for enhancing the execution performance of pipeline applications executing on the allocated grid resources (processors and network links) [13][14]. The PEPA workbench is used to calculate the performance feature, namely the throughput of executing pipeline applications, which is obtained when employing the modeled mapping for scheduling the pipeline applications onto grid resources [13][14].

Another important performance measure is the response time, which is prominently used for analyzing the performance of a resource allocation or a task scheduling system in parallel and distributed computing, where applications are bound by time constraints (such as an execution deadline). A research direction towards evaluating response time profiles has been given in the work performed

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

on evaluating PEPA models via an ordinary differential equation (ODE) analysis [14]. The functionality for evaluating response time profiles via passage time analysis has been implemented in the Imperial PEPA Compiler (IPC) tool [18], and in the PEPA workbench [16]. However, to the best of our knowledge, performance modeling for evaluating the *response times* of resource allocations in parallel and distributed computing systems and their related robustness analysis remained an open problem until the authors introduced the first solution by utilizing PEPA for analyzing the robustness of static resource allocations in [22][23].

3 Methodology: Performance Modeling using PEPA for Robustness Analysis

The target applications for this study are considered to be independent parallel applications waiting in a job queue for execution. Definitions of the notations used in the following description of the methodology are given in Table 1.

Notation	Definition
A	set of parallel applications
a_i	a parallel application $\in A$
λ_i	initial workload for a_i
$\hat{\lambda}_i$	perturbed/varied workload for a_i at runtime, where $\lambda_i \neq \hat{\lambda}_i$
M	a set of parallel machines
M_j	a parallel machine $\in M$
η_j	computational availability of machine M_j
$\hat{\eta}_j$	perturbed/varied computational availability of M_j at runtime, where $\eta_j \neq \hat{\eta}_j$
β_i^{max}	user defined makespan goal for a_i
$F_i(M_j(\hat{\eta}_j), \hat{\lambda}_i)$	Finishing time of machine M_j
ψ	robustness value of a mapping. See Equation 1
T_p	parallel execution time for a mapping calculation using PEPA passage time analysis
T_{ij}	actual time to compute a_i on M_j. Calculated as $\hat{\lambda}_i \times \hat{\eta}_j$
r_i	ideal activity rate defined as $\frac{\lambda_i}{T_{ij}}$, where $\hat{\lambda}_i = \lambda_i$
p_i	perturbed activity rate defined $\frac{\lambda_i}{T_{ij}}$, where $\hat{\lambda}_i \neq \lambda_i$

Table 1: Table of Notations

Each application receives a data set generated by three heterogeneous sensors that produce the workloads (λ_1, λ_2, and λ_3). The underlying computational system consists of parallel machines (that contain K heterogeneous processors, where K varies from machine to machine). Each machine has an associated availability factor (η_i), which is the computational availability of the allocated

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

machine for executing an application. Further, a set of possible resource allocations are considered to be available for an initial mapping of applications to machines based on an *expected time to compute* (ETC) matrix. In general, in a parallel and distributed computing environment, it is realistic to assume that the ETC values of applications on all the available machines are known a priori. Often, these estimates are derived from application profiling and machine benchmarking, from the previous executions of an application on a machine, or are provided by the user [19][20]. All applications in the job queue are assumed to start executing at time $t = 0$ seconds. In this work, the perturbation parameter considered for the robustness analysis of resource allocations is defined as the *compound effect* of perturbations generated from runtime variations in both, application workload and processor availability of a machine.

Given, A: a set of parallel applications, $a_i \in A$: a parallel application, λ_i: the workload of application a_i, β_i^{max}: a user defined makespan goal for a_i, M: a set of parallel machines, $M_j \in M$: the machine allocated to a_i, $\hat{\lambda}_i$: a perturbation parameter defined as the workload variation from the initial workload (λ_i) for an application a_i, $\hat{\eta_j}$: a perturbation parameter defined as the machine availability variation from the initial machine availability (η_j) for an application a_i, $F_i(M_j(\hat{\eta_j}), \hat{\lambda}_i)$: the finishing time of application a_i on machine M_j, then the *robustness* (ψ) of a mapping is formulated as in Equation 1.

$$\psi = \min_{\forall i \in A} \Pr[F_i(M_j(\hat{\eta_j}), \hat{\lambda}_i) \leq \beta_i^{max}] \tag{1}$$

The goal of the robustness analysis is to find a resource allocation that maximizes the robustness of the execution of the applications on the allocated parallel machines. An example PEPA model of a mapping system for two applications (a_1, a_2) and five processors $(P_0, P_1, P_2, P_3, P_4)$ distributed among two machines (M_1, M_2) has been described below. For reference, a detailed description of the PEPA language and the language operators can be found in [17][5].

$$\begin{aligned}
a_1 &\stackrel{def}{=} (compute_1, \top).RETURN \\
a_2 &\stackrel{def}{=} (compute_2, \top).RETURN \\
P_0 &\stackrel{def}{=} (compute_1, r_1).RETURN \\
P_1 &\stackrel{def}{=} (compute_1, r_1).RETURN \\
P_2 &\stackrel{def}{=} (compute_2, r_2).RETURN \\
P_3 &\stackrel{def}{=} (compute_2, r_2).RETURN \\
P_4 &\stackrel{def}{=} (compute_2, r_2).RETURN \\
M_1 &\stackrel{def}{=} P_0 \parallel P_1 \\
M_2 &\stackrel{def}{=} P_2 \parallel P_3 \parallel P_4 \\
Mapping &\stackrel{def}{=} (a_1 \parallel a_2) \underset{\mathcal{L}}{\bowtie} (M_1 \parallel M_2)
\end{aligned}$$

where $\mathcal{L} = \{compute_1, compute_2\}$

The two applications (a_1 and a_2) engage in their compute activities. In the above model, the first two statements model the two applications when they are not allocated to any processor. $\top$ is a predefined symbol in PEPA that denotes

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

an unknown rate for an activity. This symbol is used in situations when a system is carrying out some action (or sequence of actions) whose rate is unavailable at the given time. Later when the processors are modeled, a processor engages in the compute activities of all the applications mapped to this processor. At this point, the unknown activity rate ($\top$) is converted to actual activity rates (r_1 and r_2 in our example). The rate (r_1 or r_2) of the *compute* activity is calculated as a function of the speed of the processors in the machine allocated to the application and the workload for that application.

The PEPA model is provided as an input to the PEPA workbench [16] in a `*.pepa` file format. Each application and machine PEPA component in the model is translated into an underlying mathematical Markovian model by the PEPA workbench. A more detailed description of the functionality of the PEPA workbench for our study can be found in [21]. The robustness of the modeled resource allocation is obtained as a probability of attaining a predefined makespan value, which is calculated by a *passage time analysis* [18] of the computational activities of all the machine components in the generated Markovian model. The passage time analysis generates a Cumulative Distribution Function (CDF) of the passage time (T_p) from a source state (S_s) into a non-empty set of target states (S_T), such that,

$$T_p = \inf\{u > 0 : S_s(u) \in S_T | S_s(0) = \text{initial state}\} \tag{2}$$

The CDF is generated by a convolution of the state holding times over all possible paths from state $i \in S_s$ into any of the states in the set S_T [18]. For the passage time analysis, the *stop time* is analogous to the user specified makespan goal, β_i^{max}. The solution is obtained as a cumulative distribution function (CDF) of the probability of machine finishing times for the modeled resource allocation. The robustness, as formulated in Equation 1, of the resource allocation is obtained from the generated CDFs, as the minimum probability of achieving a user defined makespan goal.

4 Results and Analysis: Robustness Analysis via Numerical Modeling of Resource Allocations

In this study, a number of PEPA models have been generated for several feasible mappings of applications onto parallel heterogeneous machines. The variations in the workload values are sampled from a *uniform probability distribution* for all the applications. For this study, the Python function, `random.uniform(a,b)` has been used, with different values of `a` and `b` resulting in different mappings for a given number of applications and machines. The variations in application workload and machine availability can be sampled from any probability distribution model that can capture the static features of real workloads and machine characteristics at a particular point in time.

In PEPA, a *choice* operator permits a model component to exist in one of the many possible states of the component [5]. For example, for a component

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

$P := P1 + P2$, P can only exist in state $P1$ or $P2$. In this work, the left hand side of the + operator models the ideal execution scenario in the absence of perturbations in the sensor workload and the machine availabilities, where $\hat{\lambda_i} = \lambda_i, \forall i \in \{1, 2, 3\}$ and $\hat{\eta_j} = \eta_j$, for all machines. The right hand side of the + operator models the perturbed execution scenario generated by a compound effect of variations in application workload and machine availabilities. The compound effect is modeled as simultaneous equal variations in one or more sensor workload(s) across all applications, where $\hat{\lambda_i} \neq \lambda_i$, and equal variations in processor availability across all applications, where $\hat{\eta_j} \neq \eta_j$. The rates of the *compute* activities are calculated as a function of λ_i and the actual computation times T_{ij} of each application on the machine where it is mapped. Further, T_{ij} is calculated as a function of the estimated variation in application workload $(\hat{\lambda_i})$ and the estimated variation in the machine availabilities $(\hat{\eta_j})$, as defined in Table 1. The values of initial rates $(r_1, r_2, \cdots)$, which remain constant during runtime, are calculated as a function of the initial machine availability factor (η_j), as described later in Section 4.1. The values of perturbed rates $(p_1, p_2, \cdots)$, which vary during runtime, are calculated as a function of the varied application workload $(\hat{\lambda_i})$ and the varied machine availability factor $(\hat{\eta_j})$, as described later in Section 4.2.

The modeling generated for this study represents a resource allocation system that has a higher load imbalance factor at runtime when compared to the analysis in our prior work in [21][22][23][24]. Herein, the authors have analyzed the robustness of several mappings to study the impact of heightened load imbalance in computational systems with (i) 16 applications and 4 machines, (ii) 20 applications and 5 machines (for validation against our prior work), and (iii) 32 applications and 8 machines. Due to the state space explosion limitation of the CTMC analysis in PEPA, the authors successfully modeled only systems of less than 8 machines. Research work is ongoing to explore different implementations of PEPA that will improve the scalability of the numerical modeling and analysis of the robustness of resource allocations for larger PDC systems and applications.

4.1 Deriving PEPA activity rates in an ideal computing environment ($\hat{\lambda_i} = \lambda_i$ and $\hat{\eta_j} = \eta_j$)

The calculation for the rate is given in Equation 3, where T_{ij} is the actual time to compute an application i on machine j with initial availability (η_j) that remains constant at runtime, and λ_i is calculated as a function of the initial sensor loads $\lambda_1, \lambda_2, \lambda_3$.

$$r_i = \frac{\lambda_i}{T_{ij}} \quad \forall i, j, \; where \; T_{ij} = \lambda_i \times \eta_j \tag{3}$$

T_{ij} is calculated as a product of the runtime workload for that application (λ_i), and the machine availability factor (η_j). Rate r_i is calculated as a ratio of the application workload λ_i and T_{ij}. In the ideal execution scenario, $\hat{\lambda_i} = \lambda_i, \forall i \in$

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

$\{1, 2, 3\}$. Therefore, T_i is equal to the initial ETC values and consequently, the rates, $(r_1, r_2, \cdots)$, are only calculated using the initial machine availability (η_j).

4.2 Deriving PEPA activity rates in a perturbed computing environment ($\hat{\boldsymbol{\lambda}} \neq \boldsymbol{\lambda}$ and $\hat{\boldsymbol{\eta}}_j \neq \boldsymbol{\eta}_j$)

The calculation of the perturbed rate is given in Equation 4, where T_{ij} is the actual time to compute an application i on machine j with the varied runtime availability $(\hat{\eta}_j)$, and $\hat{\lambda}_i$ is calculated as a function of the varying sensor loads $\hat{\lambda_1}, \hat{\lambda_2}, \hat{\lambda_3}$, and λ_i is calculated as a function of the initial sensor loads $\lambda_1, \lambda_2, \lambda_3$.

$$p_i = \frac{\lambda_i}{T_{ij}} \quad \forall i, j, \text{ where } T_{ij} = \hat{\lambda}_i \times \hat{\eta}_j \tag{4}$$

T_{ij} is calculated as a product of the estimated runtime application workload $(\hat{\lambda})$, and the runtime machine availability factor $(\hat{\eta}_j)$. Henceforth, p_i is calculated as a ratio of the initial application workload λ_i and T_{ij}.

4.3 Numerical Analysis of Performance Modeling of Resource Allocations using the PEPA Workbench

The `*.pepa` model file is compiled and solved using the PEPA workbench tool. The state space derived using the Markovian analysis provided by the PEPA workbench represents the continuous time Markov chain (CTMC) processes for each component of the modeled resource allocation.

Once the state space of all the components (CTMC processes of applications and machines) of the PEPA model are generated, the tool allows the modeler to specify the type of Markovian analysis that needs to be used for solving the generated Markov models to derive performance measures [5][21]. In this work, we choose the *passage time analysis* that is used to solve the Markov models using the timing information associated with the activity rates to derive performance measures, such as, makespan and response time [18]. In this study, the passage time analysis of the Markov models for the resource allocations yields CDFs of the machine finishing times, $F_i(M_j(\hat{\eta}_j), \hat{\lambda}_i)$, as passage times between the states associated with applications assigned to a machine. An example of the CDF of the finishing time of machine M_1 in one of our randomly generated resource allocation model is shown in Figure 2. A comparison of the finishing times of each machine with respect to the four mappings (selected from a larger set of generated mappings due to space constraints) is illustrated in Figure 3 for two different execution scenarios, (i) 16 applications and 4 heterogenous machines, and (ii) 20 applications and 5 heterogenous machines.

A key benefit resulting from the performance modeling and analysis in this work is to identify the most robust resource allocation among the ones delivering equal execution performance. The robustness metric ψ_x is calculated using Equation 5. For this study, the makespan goal is set as $\beta_i^{max} = 45$ seconds.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

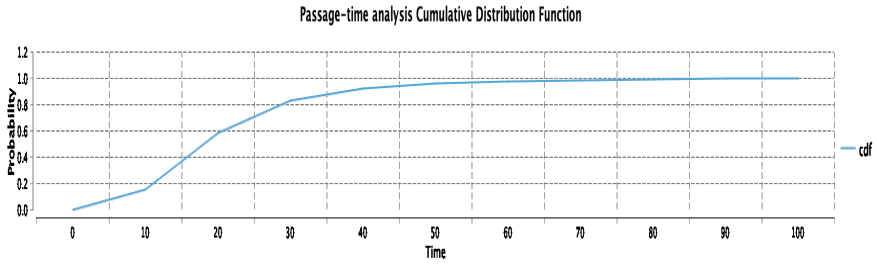
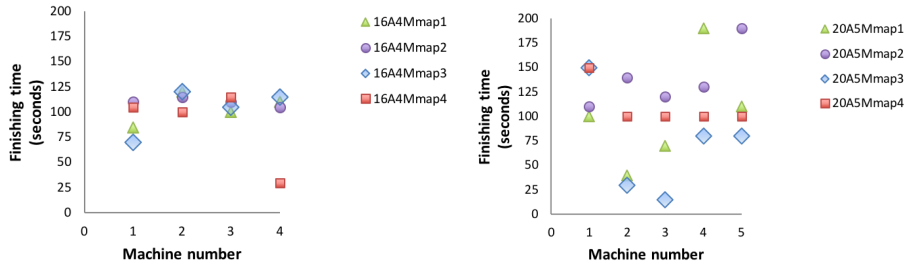


Fig. 2: Cumulative distribution function (CDF) of the finishing time of machine M_1 for a resource allocation mapping 20 applications onto 5 machines.



(a) (b)

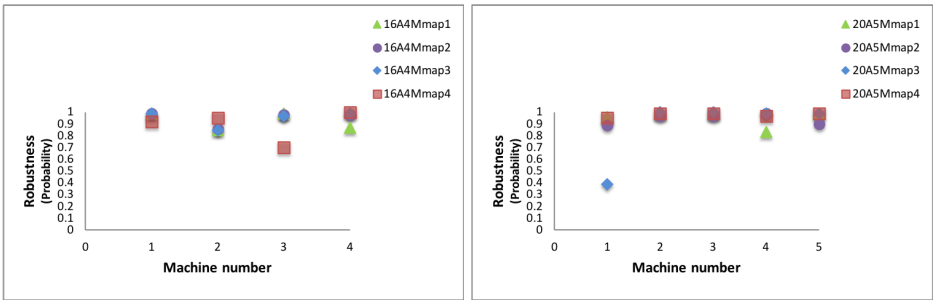
Fig. 3: A comparative analysis of the finishing times of each machine derived from the execution of 4 different mappings of (a) 16 applications onto 4 heterogeneous machines, and (b) 20 applications onto 5 heterogeneous machines

$$\psi_x = \min_{\forall i,j \ in \ Mapping_x,} \Pr[F_i(M_j(\hat{\eta}_j), \hat{\lambda}_i) \leq 45] \tag{5}$$

The robustness value for each machine is calculated as the probability of that machine to finish before the makespan goal. A comparison of the robustness values of all the machines for each of the four mappings of (i) 16 applications to 4 heterogeneous machines, and (ii) 20 applications to 5 heterogeneous machines is illustrated in Figure 4.

The results illustrated in Figures 5a and 5b indicate that for both test cases of resource allocations modeled, one for a PDC system with 16 applications and 4 heterogeneous machines, and the other for a PDC system with 20 applications and 5 heterogeneous machines, resource allocations promising better parallel execution time over the others can be identified. Moreover, the faster resource allocations that deliver equal execution makespan also differ vastly in their degrees of robustness. For example, in the case of mappings `20A5Mmap3` and `20A5Mmap4` in Figure 5b, both mappings deliver equal execution performance in terms of system makespan. However, Mapping `20A5Mmap4` is 2.43 times more ro-

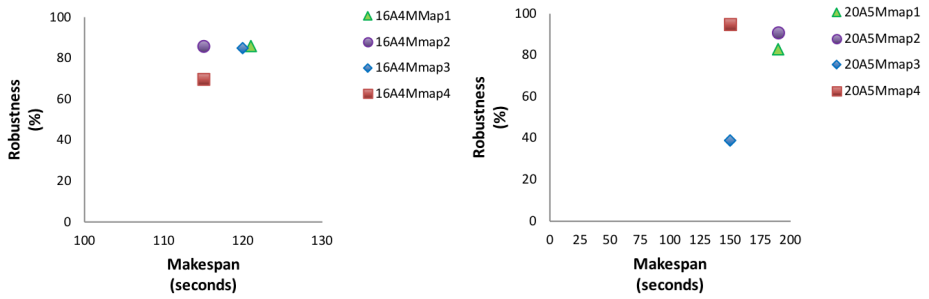
ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44



(a) (b)

Fig. 4: A comparative analysis of the robustness values obtained for each machine for the four mappings of (a) 16 applications onto 4 heterogeneous machines, and (b) 20 applications onto 5 heterogeneous machines. The makespan goal is set as $\beta_i^{max} = 45$ seconds.

bust that Mapping `20A5Mmap3`. Therefore, the value of robustness for `20A5Mmap4` significantly increases by 143% over `20A5Mmap3`. For highly critical applications, this substantial improvement in robustness, enables a more informed decision for selecting the most appropriate mapping that can withstand the runtime perturbations in application and system parameters in a parallel computing environment, and can guarantee the best execution performance. For example, in this modeling study, Mapping `20A5Mmap4` is a *more robust* choice for an initial allocation in terms of achieving a set makespan goal in the presence of runtime perturbations caused by a compound effect from variations in application workload and machine availability.



(a) (b)

Fig. 5: A comparison between the robustness values and the performance in terms of the system makespan of the four mappings of (a) 16 applications onto 4 heterogeneous machines, and (b) 20 applications onto 5 heterogeneous machines.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

5 Conclusions and Future Work

The analytical study performed in this work to obtain the robustness of *static* resource allocations, which are analytically modeled and numerically solved using PEPA, provides a useful tool that can be incorporated in the design phase of a resource allocation system in a parallel and distributed computing environment. Although the analysis is limited to *static* resource allocations, a key benefit of this study is learning that, in the case of resource allocations promising equal execution performance, the numerical evaluation of the PEPA models enables a selection of the most robust resource allocation in the presence of runtime perturbations caused by a *compound effect* arising from dynamic variations in *application workload* and *machine availability*. For each of the several modeled resource allocations, a numerical analysis of the PEPA models yields the execution performance as the parallel execution time. The robustness value for each mapping is calculated as the minimum probability of the machines to finish before a desired makespan goal. The results presented in Section 4 indicate that two or more different mappings yield equal execution performance. However, a significant improvement (up to 143%) in the robustness value among the mappings yielding equal parallel execution times has been observed. Thus, this substantial increase in robustness strongly suggests the selection of one mapping versus the other for *guaranteeing* the best execution performance.

To the best of our knowledge, this work is the first effort towards modeling the execution of applications on heterogeneous machines by simultaneously incorporating variations in both application workload and machine availabilities, considered together as a *compound perturbation*. To facilitate and ease the reproducibility of our research, we provide a Singularity container of the PEPA Workbench that has been validated to produce identical results to the non-containerized version of the PEPA Workbench [25]. The PEPA model and build recipe for the Singularity container are available at https://github.com/williamssanders/pepa, and the container is publicly available at https://www.singularity-hub.org/collections/2351.

In the future, the authors plan to continue this study towards performance modeling of resource allocations and numerical evaluation of their robustness on larger scale PDC systems and data sets from real scientific applications. To facilitate a scalable evaluation, the authors plan to investigate a number of other existing implementations of PEPA that are not limited by state space explosion. Further, the authors also plan to explore an integration of the PEPA models into a runtime scheduler/controller in a model-based framework, where the embedded models can be re-evaluated with minimal overhead when a system parameter changes at runtime.

Acknowledgment

The authors would like to acknowledge The Jackson Laboratory for providing support and helping to make this work possible. Additionally, the authors would like to acknowledge the partial support of the NSF #1034897 grant.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

References

1. S. Ali, A. Maciejewski, H. Siegel, and J.K. Kim, *Measuring the robustness of a resource allocation*. IEEE Transactions on Parallel and Distributed Systems, vol. 15, no. 7, pp. 630 - 641, 2004.
2. S. Ali, J.-K. Kim, H. J. Siegel, and A. A. Maciejewski, *Static heuristics for robust resource allocation of continuously executing applications*. J. Parallel Distrib. Comput., vol. 68, no. 8, pp. 1070 - 1080, Aug. 2008.
3. I.Banicescu, F.M.Ciorba, and R.L.Carino, *Towards the robustness of dynamic loop scheduling on large-scale heterogeneous distributed systems*. In Proceedings of the IEEE International Symposium on Parallel and Distributed Computing (ISPDC 2009), pp. 129 - 132, 2009.
4. S. Srivastava, I. Banicescu, and F. Ciorba, *Investigating the robustness of adaptive dynamic loop scheduling on heterogeneous computing systems*. In Proceedings of The 2010 IEEE/ACM International Symposium on Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW-PDSEC, On CD-ROM), pp. 1 - 8, April 2010.
5. J. Hillston, *A Compositional Approach to Performance Modelling*. Cambridge University Press, 1996.
6. E. Coffman and J. Bruno, *Computer and job-shop scheduling theory*. A Wiley-Interscience publication. Wiley, 1976.
7. D.Fernandez-Baca, *Allocating modules to processors in a distributed system*. IEEE Trans. Softw. Eng., vol. 15, no. 11, pp. 1427 - 1436, Nov. 1989.
8. O. H. Ibarra and C. E. Kim, *Heuristic algorithms for scheduling independent tasks on nonidentical processors*. J. ACM, vol. 24, no. 2, pp. 280 - 289, Apr. 1977.
9. J. V. Leon, D. S. Wu, and R. H. Storer, *Robustness Measures and Robust Scheduling for Job Shops*. IIE Transactions, vol. 26, no. 5, pp. 32 - 43, 1994.
10. R. L. Daniels and J. E. Carrillo, *Beta-robust scheduling for single machine systems with uncertain processing times*. IIE Transactions, vol. 29, no. 11, pp. 977 - 985, 1997.
11. S. Gertphol and V. Prasanna, *Mip formulation for robust resource allocation in dynamic real-time systems*. In Proceedings of the International Parallel and Distributed Processing Symposium, 2003.
12. S. Gertphol and V. Prasanna, *Iterative integer programming formulation for robust resource allocation in dynamic real-time systems*. In Proceedings of the International Parallel and Distributed Processing Symposium, 2004.
13. A. Benoit, M. Cole, S. Gilmore, and J. Hillston, *Evaluating the performance of skeleton-based high level parallel programs*. Computational Science-ICCS 2004. Springer, 2004, pp. 289 - 296.
14. A. Benoit, M. Cole, S. Gilmore, and J. Hillston, *Scheduling skeleton-based grid applications using pepa and nws*. The Computer Journal, vol. 48, no. 3, pp. 369 - 378, 2005.
15. V.L.Wallace and R.S.Rosenberg, *Markovian models and numerical analysis of computer system behavior*. In Proceedings of the Spring joint computer conference. ACM, 1966, pp. 141 - 148.
16. S. Gilmore and J. Hillston, *The pepa workbench: A tool to support a process algebra based approach to performance modelling*. In Computer Performance Evaluation Modelling Techniques and Tools, ser. Lecture Notes in Computer Science, G. Haring and G. Kotsis, Eds. Springer Berlin Heidelberg, 1994, vol. 794, pp. 353 - 368.
17. J. Hillston, *Tuning systems: From composition to performance*. The Computer Journal, vol. 48, no. 4, pp. 385 - 400, May 2005.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44

18. J. T. Bradley, N. J. Dingle, S. T. Gilmore, and W. J. Knottenbelt, *Derivation of passage-time densities in pepa models using ipc: The imperial pepa compiler*, 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer Telecommunications Systems, MASCOTS 2003. pp. 344 - 351.
19. A. Ghafoor and J. Yang, *A distributed heterogeneous supercomputing management system*. Computer, vol. 26, no. 6, pp. 78 - 86, June 1993.
20. M. A. Iverson, F. Ozguner, and L. Potter, *Statistical prediction of task execution times through analytic benchmarking for scheduling in a heterogeneous environment*. IEEE Trans. Comput., vol. 48, no. 12, pp. 1374 - 1379, Dec. 1999.
21. Srishti Srivastava, *Evaluating the robustness of resource allocations obtained through performance modeling with stochastic process algebra*. PhD dissertation, Mississippi State University, Department of Computer Science and Engineering, May, 2015.
22. I. Banicescu and S. Srivastava, *Towards Robust Resource Allocations via Performance Modeling with Stochastic Process Algebra*. In Proceedings of the 18th IEEE International Conference on Computational Science and Engineering (CSE-2015), pp. 270 – 277, Porto, Portugal, October 2015.
23. Srivastava, S., and Banicescu, I. (2017) *Robust resource allocations through performance modeling with stochastic process algebra*. Concurrency and Computation: Practice and Experience, vol. 29(7): e3894. doi: 10.1002/cpe.3894.
24. S. Srivastava and I. Banicescu, *PEPA Based Performance Modeling for Robust Resource Allocations Amid Varying Processor Availability*. In proceedings of the 17th IEEE International Symposium on Parallel and Distributed Computing (ISPDC), Geneva, 2018, pp. 61-68.
25. W.S. Sanders, S. Srivastava, and I. Banicescu. *A Container-Based Framework to Facilitate Reproducibility in Employing Stochastic Process Algebra for Modeling Parallel Computing Systems*. In proceedings of the 2019 IEEE/ACM International Symposium on Parallel Distributed Processing, Workshops and PhD Forum (IPDPSW-HIPS), in press.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50417-5_44