2-9-2011

Theory of Resource Allocation for Robust Distributed Computing

Jorge E. Pezoa

Jorge E. Pezoa
Candidate

Electrical and Computer Engineering
Department

This dissertation is approved, and it is acceptable in quality and form for publication:

Approved by the Dissertation Committee:

Majeed Hayat, Chairperson

Wasin Kharé

Peter Loh

Sattar B

Theory of Resource Allocation for Robust Distributed Computing

by

Jorge E. Pezoa

B.E., Electronics Engineering,
Universidad de Concepción, Chile, 1999
M.S., Electrical Engineering,
Universidad de Concepción, Chile, 2003

DISSERTATION

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctor of Philosophy
Engineering

The University of New Mexico

Albuquerque, New Mexico

September, 2010

Dedication

To my "amorcito" Angely, to my beloved parents, and to the memory of my uncles: Mario, Carlos and Arisnel.

Acknowledgments

I would like to gratefully and sincerely thank my advisor Prof. Majeed M. Hayat for his guidance, encouragement, support, friendship, kindness, and help during all these years. It has been a pleasure and also an honor to work under his guidance. I also appreciate the opportunities he gave me to get involved in different projects. These opportunities have largely improved my education and have become invaluable tools in my professional life. Further, I would like to thank the financial support that he gave me during my graduate study. I would like to thank also to his lovely family, Karen, Kari and Allison, for their love, support and all the wonderful times and memories we have shared during these years in Albuquerque.

I thank also Drs. Yasamin Mostofi, Patrick Bridges, Nasir Ghani and Balu Santhanam for accepting to be part of my dissertation committee, for their help during the development of this work, and also for the excellent classes they lecture and I had the pleasure to take. I must thank Dr. Sagal Dhakal, Mrs. Mahshid Rahnama, Mr. Alejandro González, and Mr. Zhuoyao Wang for the enormous amount of help they provided me in this work. Special thanks also to my friends at UNM and Albuquerque: Biliana & Pavel, David, Luciana & Germán, Lissette & Saúl, and Carmen.

I would like to thank also the help and support received from The UNM Center for Advanced Research Computing. I specially thank Dr. Susan Atlas, Center Director, Dr. Timothy Thomas, Deputy Director, and Mr. Jim Prewett, Systems Engineer, which gently provided the cluster `ristra` for conducting experiments and calculations related to this work.

At last, I would like to thank my beloved wife Angely, my parents Rosa and Edgardo, my brother Jaime, my aunts Carmen, Eliana and Elsa, my parents-in-law Rosa and René, and my entire family for their love, unconditional support, patience, and good vibrations. Special thanks to Rayitas for her love and the great company she provides. Thank all of you very much!

This work has been supported by Prof. Majeed M. Hayat's National Science Foundation Information Technology Research (ITR) Grant No. ANI-0312511 and by the Defense Threat Reduction Agency (Combating WMD Basic Research Program).

Theory of Resource Allocation for Robust Distributed Computing

by

Jorge E. Pezoa

ABSTRACT OF DISSERTATION

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctor of Philosophy
Engineering

The University of New Mexico

Albuquerque, New Mexico

September, 2010

Theory of Resource Allocation for Robust Distributed Computing

by

Jorge E. Pezoa

B.E., Electronics Engineering,

Universidad de Concepción, Chile, 1999

M.S., Electrical Engineering,
Universidad de Concepción, Chile, 2003

Ph.D., Engineering, University of New Mexico, 2010

Abstract

Lately, distributed computing (DC) has emerged in several application scenarios such as grid computing, high-performance and reconfigurable computing, wireless sensor networks, battle management systems, peer-to-peer networks, and donation grids. When DC is performed in these scenarios, the distributed computing system (DCS) supporting the applications not only exhibits heterogeneous computing resources and a significant communication latency, but also becomes highly dynamic due to the communication network as well as the computing servers are affected by a wide class of anomalies that change the topology of the system in a random fashion. These anomalies exhibit spatial and/or temporal correlation when they result, for instance,

from wide-area power or network outages These correlated failures may not only inflict a large amount of damage to the system, but they may also induce further failures in other servers as a result of the lack of reliable communication between the components of the DCS. In order to provide a robust DC environment in the presence of component failures, it is key to develop a general framework for accurately modeling the complex dynamics of a DCS.

In this dissertation a novel approach has been undertaken for modeling a general class of DCSs and for analytically characterizing the performance and reliability of parallel applications executed on such systems. A general probabilistic model has been constructed by assuming that the random times governing the dynamics of the DCS follow arbitrary probability distributions with heterogeneous parameters. Auxiliary age variables have been introduced in the modeling of a DCS and a hybrid continuous and discrete state-space model the system has been constructed. This hybrid model has enabled the development of an age-dependent stochastic regeneration theory, which, in turn, has been employed to analytically characterize the average execution time, the quality-of-service and the reliability in serving an application. These are three metrics of performance and reliability of practical interest in DC. Analytical approximations as well as mathematical lower and upper bounds for these metrics have also been derived in an attempt to reduce the amount of computational resources demanded by the exact characterizations.

In order to systematically assess the reliability of DCSs in the presence of correlated component failures, a novel probabilistic model for spatially correlated failures has been developed. The model, based on graph theory and Markov random fields, captures both geographical and logical correlations induced by the arbitrary topology of the communication network of a DCS. The modeling framework, in conjunction with a general class of dynamic task reallocation (DTR) control policies, has been used to optimize the performance and reliability of applications in the presence of in-

dependent as well as spatially correlated anomalies. Theoretical predictions, Monte-Carlo simulations as well as experimental results have shown that optimizing these metrics can significantly impact the performance of a DCS. Moreover, the general setting developed here has shed insights on: (i) the effect of different stochastic models on the accuracy of the performance and reliability metrics, (ii) the dependence of the DTR policies on system parameters such as failure rates and task-processing rates, (iii) the severe impact of correlated failures on the reliability of DCSs, (iv) the dependence of the DTR policies on degree of correlation in the failures, and (v) the fundamental trade-off between minimizing the execution time of an application and maximizing its reliability.

Contents

List of Figures

List of Tables

Glossary

CS computing server

DC distributed computing

DCS distributed computing system

DES discrete event simulation

DTR dynamic task reallocation

FN failure-notice

HP high-performance

LB load balancing

MC Monte-Carlo

MRF Markov random field

P2P peer-to-peer

pdf probability distribution function

QoS quality-of-service

tps tasks per second

TR task reallocation

WMD weapons of mass destruction

WSN wireless sensor network

Chapter 1

Introduction

Parallel computing is defined as the simultaneous execution of an application on multiple processors. Applications can be executed in a parallel fashion only if they can be partitioned into small independent tasks. After such partitioning process, tasks must be allocated onto the processors, and only after this process the application can starts its execution in parallel. The goal in parallel computing is to achieve the results faster as compared to executing the same application on a single processor.

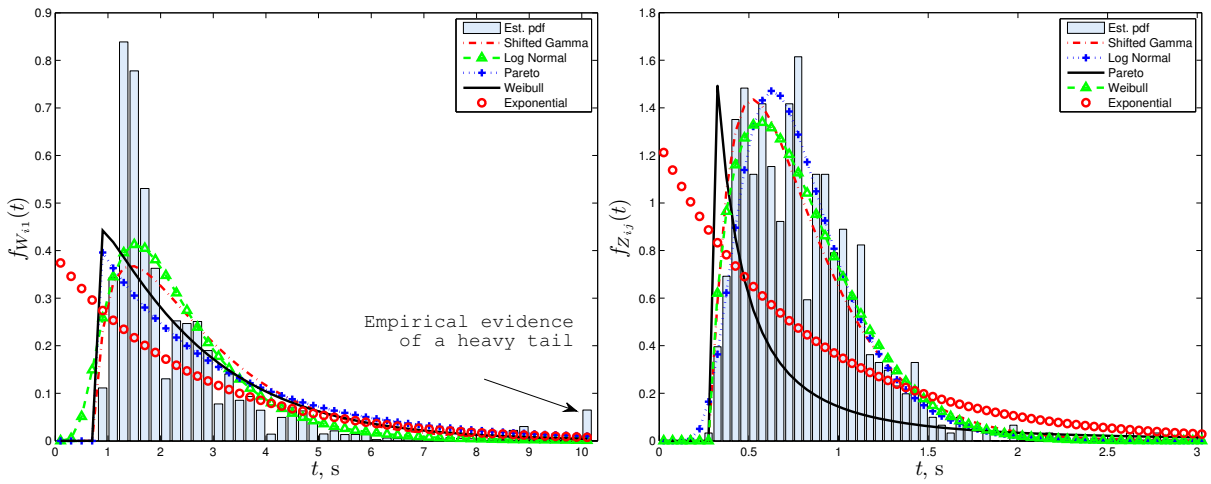
The type of parallel computing where computing servers (CSs) do not have access to a shared memory and must communicate each other by means of a network is called distributed computing (DC). The popularity of DC has increased because it allows the execution of large, computationally intensive parallel applications at an inexpensive cost. This inexpensive cost comes from the fact that DC is typically performed on clusters of low-end workstations interconnected by a network that exhibits both low bandwidth and a significant latency to any exchange of the information. Thus, unlike parallel computing systems, distributed computing systems (DCSs) are multi-computer environments, with a distributed-memory, heterogeneous computing capabilities and non-negligible communication costs.

In DC there is a fundamental trade-off between the execution time of an application and the communication costs between the processors. The performance of applications executed on DCSs can be improved, at runtime, by migrating the tasks among the servers. In DCSs, effective dynamic task reallocation (DTR) strategies must account for the heterogeneous capabilities of the servers as well as for the transfer delays imposed by the communication network. Unfortunately, practical DCSs suffer from component failures that may halt the execution of applications; therefore, providing a robust DC environment to the users becomes essential. Improving the robustness in a DCS is a challenging problem because DTR policies must trade off reliability, communication costs and execution time, while simultaneously considering the heterogeneous capabilities of the system components. Trading off communication costs, application's reliability and DCS performance is the gist of this dissertation, and a novel mathematical framework for optimizing performance and reliability in DCSs in the presence of component failures is the major contribution of this work.

Lately, DC has emerged in applications such as grid computing [23], reconfigurable and high-performance (HP) computing, wireless sensor networks (WSNs), distributed pattern-searches in DNA databases [82], battle management systems [19], surveillance and threat detection, peer-to-peer (P2P) networks, and donation grids [78]. When DC is executed in these applications, the DCS not only offers to its users heterogeneous computing resources and a significant communication latency, but also exhibits a topology that changes over time in a random fashion due to such applications are affected by a wide class of anomalies that may exhibit spatial and/or temporal correlations. Consider, for instance, the case of a cluster of servers in a DCS that are geographically or logically coupled. These servers may fail simultaneously due to a wide-area power or network outage. In addition, consider now the case of DC in WSNs, the topology of the DCS may vary due to a coordinated action executed by the sensors in order to save batteries. Such kind of energy-preserving action typically involve turning off a subset of the sensor nodes. In battle management systems, for

example, real-time DC is performed in harsh environments and massive disruptions can result from attacks using weapons of mass destruction (WMD). In fact, the arising of real-time DC in harsh environments has triggered government agencies, such as the Defense Threat Reduction Agency, to launch research initiatives in network science to understand the extent of damage that can be inflicted upon networks in the event of attacks and also to develop strategies to increase the robustness of networks when a threat is present. Since performance and reliability of applications are known to be highly dependent on the stochastic attributes of component failures, it is plausible to conjecture that correlated failures have more adverse effects on the performance and reliability than independent failures. These emerging scenarios for DC have generated new challenges where techniques developed for traditional DCSs are no longer appropriated. *This new paradigm for DC calls for novel models for DCSs as well as new assessment tools that can adapt to both workload fluctuations and changes in the number of available CSs.*

In order to provide a robust DC environment in the presence of network anomalies, it is key to develop a general framework for accurately predicting the performance and reliability of the applications being executed on a DCS. This general framework must capture the heterogeneity and randomness in both the processing capabilities of the servers and in the communication network. In addition, such a general framework must include the type of correlation exhibited by the anomalies altering the topology of the system, and must account also for a DTR policy so that performance and reliability of applications can be improved at no extra cost to the system. To the best of the author's knowledge, such framework at the level of generality stated has not been proposed heretofore.



(a) (b)

Figure 1.1: Empirical pdfs and several fitted distributions for: (a) task execution time at a server; and (b) task transfer-time between two servers communicating over a wireless channel.

1.1 Motivation

Modeling DCSs and assessing their performance and reliability are complicated tasks due to the fact DCSs comprise a large number of elements that interplay in a concurrent and stochastic manner, thereby creating a complex system dynamics [11, 22, 23]. Because of this complexity, most of the analytical approaches used to analyze performance and reliability introduce the simplifying assumption about the Markovian behavior of the DCS. Under the Markovian assumption, all the concurrent events governing the behavior of the DCS are assumed to follow exponential distributions [4, 9, 42, 47, 51, 67].

Unfortunately, the Markovian assumption is not always appropriate to model real systems. For example, Figs. 1.1(a) and (b) show, respectively, the empirical pdfs of the service time of tasks and the transfer times of tasks. These pdf have been ob-

tained after processing data logged from experiments conducted on the testbed DCS described in Section 4.2.3. These experiments were conducted using non-dedicated computers that communicate over an IEEE 802.11g wireless channel during working hours on a normal day of work at the third floor of the Electrical and Computing Engineer Department building. In the figures different stochastic models for approximating the random service and transfer times are presented. First, it can be noted that real systems impose unavoidable minimum response times to the service or to the transfer of a task. In the case of the service time, this minimum time results from operations conducted by the operating system and the application, such as loading the data into the memory, making system calls and initializing parameters. In the case of the transfer times, the communication networks employed by actual DCSs always introduce a non-zero end-to-end propagation delay to any exchange of information. From these physical constraints of real systems, the probability distribution of the random service and transfer times cannot be accurately modeled by an exponential distribution. Second, from the shape of the empirical pdfs, the exponential distribution does not appear as the appropriate distribution for the data. In the case of the service time, evidence of a heavy tail is observed at the point mass centered at $t = 10$ s, while in the case of the transfer time the pdf resembles a Gamma or a Log-normal distribution. In addition, from the figures it is easy to observe that the total approximation errors introduced by the exponentially fitted pdfs are larger than those introduced by the remaining probability distributions considered in the example. From these observation the following research questions arise: under which conditions the exponential models are accurate approximations for the service and transfer times? How these approximations impact the performance and reliability metrics?

In [67] the question on how much the exponential approximation affects the service reliability metric was studied by means of Monte-Carlo (MC) simulations. Preliminary results of this study indicated that the exponential model for the reliability

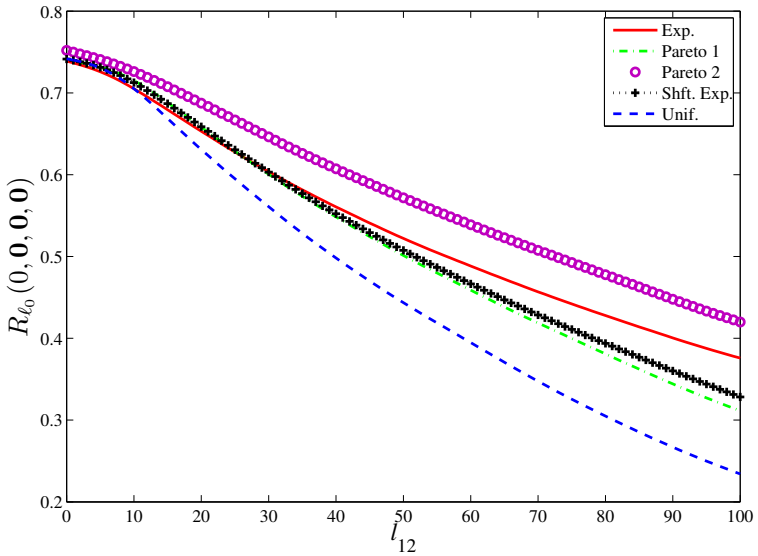


Figure 1.2: The service reliability (i.e., the probability of executing an application) of a DCS as a function of a task reallocation policy that exchanges l_{12} tasks between a pair of servers. Pareto, shifted exponential and uniform stochastic models for the task transfer times are approximated by an exponential ("Exp.") model. As the amount of tasks exchanged increases, the exponential approximation looses its accuracy in predicting the reliability.

yields accurate predictions for such metrics and the approximation errors appeared to be no larger than 4%. However, further simulations showed that as the ratio between the average transfer time of tasks and the average service time of them increases, the exponential approximation looses its accuracy in predicting the reliability. In particular, unacceptable errors of 120% were found when the ratio between these times was five. In addition, another simulation-based study confirmed that under the same operation regime, the performance metric average execution time is also affected by wrongly assuming that the random events driving the DCS dynamics follow exponential distributions. Figure 1.2 depicts the theoretical predictions for the service reliability as a function of the number of tasks exchanged among the servers, for several stochastic models for the transfer times. (This example is explained in detail in Section 3.3.2 of this dissertation.) This figure shows that when a few number of

tasks are exchanged in the network, i.e., when the communication cost is "small," the approximations provided by the exponential distribution remains very accurate for all the stochastic models considered. However, as soon as the number of tasks exchanged increases, i.e., the transfer time becomes larger, the exponential approximation is no longer valid. Specifically, for the example showed in Fig. 1.2 a maximum relative approximation error for the service reliability of 65% has been found. This approximation errors introduce another research question: does the modeling error affect the DTR policies?

From these examples, it can be observed that Markovian models for performance and reliability metrics are not suitable for representing the dynamics of general DCSs. Even though the existing Markovian approaches can yield accurate approximations for such metrics, the applicability of these models is clearly limited.

Finally, most of the reliability analysis of DCSs has been conducted heretofore assuming that failure components are independent. However, in the emergent scenarios for DC correlated may naturally occur. The presence of this type of failures immediately limitates the scope of application of the existing approaches for assessing reliability in DCSs. It was mentioned that one can conjecture about the more negative impact of correlated failures on the reliability of DCSs as compared to independent failures. However, in order to systematically assess reliability in the presence of spatially correlated failures, it is mandatory first to have tools for modeling such kind of component failures. In the literature only few analytical models for correlated failures are available and most of them have been derived from the statistical analysis of logged data. Moreover, it has been observed in the literature that there is a need for efficient simulation tools for generating patterns of failures with spatial correlation. Thus, another motivation for this dissertation is to satisfy the need for general analytical models as well as simulation tools for the assessing reliability in DCSs with arbitrarily defined topologies, when they are affected by correlated

failures.

1.2 Prior work

1.2.1 Modeling distributed computing systems

Modeling DCSs and assessing the performance and reliability of applications executed on these DCSs are complicated tasks for several reasons. First, the computing servers composing a DCS offer heterogeneous processing capabilities to their users. Second, servers in a DCS are not normally dedicated processors; the processing time of tasks at any server varies in time according to the random fluctuations of the load executed by the servers. Third, the communication network of a DCS interconnects geographically dispersed servers, and consequently, imposes a significant random and heterogeneous latency to any exchange of information. Finally, a DCS comprises a large number of elements that interplay in a concurrent and stochastic manner, thereby resulting in complex system dynamics [11, 22, 23].

Because of this unavoidable complexity, simulation, and in particular discrete event simulation (DES), has emerged as the most employed method for both modeling and conducting performance and reliability analysis in heterogeneous DCSs. The most appealing aspect of DES is its simplicity and general application, while its main drawback is its high computational cost to achieve accurate results [11]. In addition to DES, another type of theoretical modeling has been conducted by means of specification languages and graphical modeling tools such as Petri Nets [10], fault trees [12], and reliability graphs [52].

Analytical methods for performance and reliability have also been developed in the literature. To circumvent the inherent complexity of the models, researchers usually impose a set of simplifying assumptions and constraints so that system's be-

havior can be characterized in a tractable manner. The most common simplifying assumptions employed in the literature are heuristics related to applications' execution time [5, 83, 88], homogeneous capabilities of servers and/or communication links [5], time-invariant DCS topology [43], and the deterministic behavior of the transfer time of tasks [7,47,88]. Even though these assumptions are meaningful in parallel computing systems, the random nature of the uncertainty introduced by the communication network and the number of functioning servers necessitates using probabilistic models for conducting performance and reliability analysis of DCSs [10, 44, 71].

The stochastic model most widely used to represent a DCS is the Markovian distributed-queueing network. Such model is obtained under the assumption that all the concurrent events governing the behavior of the DCS follow exponential distributions [4, 9, 42, 47, 51, 67]. The extensive amount of research on Markovian models has yielded several modeling and analysis tools such as TimeNET and WebSPN [9,42,47,51]. The main advantages provided by the Markovian assumption are that highly simplifies the calculations, avoids tractability problems and the time-dependent behavior of system dynamics [62], and leads to closed-form [9, 42, 51], or recursive characterizations [28, 67], of very complex performance metrics.

Analytical modeling and analysis of queueing systems in non-Markovian settings has been also conducted. The classical approach is to abstract a DCS in terms of G/G/n queues [23,58], and more recently, in terms of non-Markovian stochastic Petri Nets [10,38,85]. Then, analysis can be conducted using basic principles and standard methods from stochastic processes such as state-space expansion based on continuous and discrete phase-type distributions [11], or based on supplementary variables [11, 20]. As a result of these works, researchers have observed that Markovian models may introduce significant errors in the calculation of performance and reliability metrics [38, 67, 70]. In particular, it is shown here that the service reliability of a DCS calculated exploiting the Markovian assumption is highly inaccurate in settings

where the average task-transfer delays are large compared to the average task service-time. These findings have been reported also in [67, 70].

In addition, we can comment that non-Markovian analysis of queueing systems has been conducted in a much simpler scenario: a $G/G/1$, i.e., in scenarios where a single server is processing the workload and the inter arrival times follow general distributions. As in the case of multiple servers, closed form solution are not simple to derive and in some cases are even impossible to obtain. Approaches like the technique of supplementary variables developed by Cox, [20], and large deviation characterizations for heavy and lightly tailed distributions have been obtained by several researchers, where the groups by Gallaler, Tsitsiklis, and Modiano at the Massachusetts Institute of Technology are one the most active in the area, [37,49,64].

1.2.2 Spatially correlated failures

It was stated that modeling and assessing the performance and reliability of applications executed on DCSs are complicated tasks. What makes these tasks even more complicated is that, in the aforementioned emerging scenarios for DC, server failures can exhibit correlation in space and/or time. To date, there are few works in the literature tackling the problem of reliability in the presence of correlated failures. In most of the work the problem has been tackled in contexts different from DC; however, the common factor is that none of them present a systematic stochastic approach to model correlated failures. For example, Fu and Xu reported in [35] a proactive failure management in networked computing systems using a failure predictor based on spatial and temporal correlation. The failure correlation has been empirically modeled in this work by analyzing system event logs and databases of failure signatures. In a similar sense, Jiang and Cybenko attempted in [50] to detect correlated failures in a computer network security system, while Tang and Iyer employ mea-

surements from a cluster to assess reliability for different types of software-induced correlated failures [81]. In [23], Dai *et al.* evaluated the reliability of a grid computing system considering the failure correlation of different subtasks executed by the grid, but component failures were still assumed to be independent. In [24, 41], a software reliability modeling framework was reported, which is based on Markov renewal processes. The framework is capable of incorporating the dependencies among successive software runs. Recently, approximate analytical expressions for reliability in on-demand systems exhibiting correlated failures has been developed in order to assess the impact of pairwise component correlations on system reliability [34].

1.2.3 Dynamic task reallocation for performance and reliability

The role of DTR in improving the performance of DCSs has been studied vastly considering a number of performance metrics; these include the average response time of an entire application [28, 73], the probability of completely serving an application [6,17,22,23,25,26,66], the probability of serving an application within a given amount of time [77], the average queue-length of a server [48, 84], and the total sum of communication and service times [54, 55]. In addition, the problem of devising task reallocation (TR) policies has been studied under both static and dynamic scenarios. In static TR, a centralized entity allocates the tasks offline, that is, tasks are allocated prior to their execution in the DCS [22, 23, 84]. In contrast, in DTR tasks are queued at the servers and DTR is triggered online whenever there is an imbalance in the DCS [28, 29, 44, 55]. Searching for the optimal solution to the problem of devising DTR policies is stated as a mixed integer optimization problem, which is known to be NP-hard [8, 32, 83]. In [8, 71] authors state that heuristic methods can give near optimal or in some cases optimal results.

DTR policies are typically employed as mechanisms to achieve an even distribution of tasks among the servers [28, 84]. This process is usually referred to as load balancing (LB). A LB policy is efficiently using the computing resources of a DCS if the application is evenly distributed among all the servers, [15, 21, 89], the amount of inter-server communication is kept small, [46, 57, 60], the cost incurred in transferring the tasks is smaller than the waiting time of the tasks in the queues of the servers, and the computing overhead incurred by the LB algorithm is small [16]. Clearly, there is an inherent trade-off between the aforementioned reduction in communication and the disseminating of load and network state information. Moreover, in DCS environments where communications costs are expensive (e.g. wireless channels) or where communication links are not reliable the communication overhead is an important issue to consider.

DTR has also been effectively employed to reduce the effect of server failures on the service of an application. The objective is to maximize the service reliability, while the service time of the application is simultaneously minimized. To date, existing analytical solutions to this problem have been based upon multi-objective optimization approaches. Some approaches have assumed deterministic communication delays [56, 61, 65, 75] while introducing task and/or hardware redundancy to compensate for the communication delays [72, 74]. Other solutions either exploit *a priori* information on the network configuration [79] or provide computationally fast solutions by using heuristic algorithms such as genetic algorithms [88] and simulated annealing [7, 43]. It is argued here that when DC is performed in scenarios where servers may fail permanently, the uncertainty introduced by both the number of functioning servers and the communication network, the DTR problem has to be tackled in a probabilistic framework. The papers most relevant to this dissertation are the works by Dai *et al.* [22,23]. The authors solve the static TR problem by using a centralized entity, which allocates tasks in the DCS in order to maximize the service reliability. In these works, the authors have considered random communication

delays as well as random server failure. Additionally, in [26] the effect of failure and recovery of servers on the average service time of an application has been studied when the DCS is composed by two servers.

Prior work by the Resource Allocation Group at UNM

In the last years, Professor Hayat's Resource Allocation Group, formerly Load Balancing Group, at UNM has conducted collaborative research with The University of Tennessee–Knoxville (UTK). The group has performed research on modeling, optimization, and testing of LB policies in large-scale DCS, [1–3, 18, 27, 28, 67]. Chiasson *et al.* developed deterministic linear and nonlinear models for DCS where the application as well as system information transfer-times are not negligible. In addition, they characterized the stability of the DCS in terms of the transfer-times of information and the partition of the application in the LB algorithm, [2, 3, 18]. Ghanem *et al.* developed and implemented a layered multiplatform DCS architecture using the C programming language, POSIX-threads, and the TCP/IP protocol stack [39, 40]. Such architecture has been used to experimentally test all the LB and the DTR policies developed by the group. The architecture is explained in the next section and the contributions made in this dissertation to the original software are commented. The experiments conducted by the group were performed over several communication networks: LANs, wireless LANs, Internet and PlanetLAB. Dhakal *et al.*,using a novel regeneration-theory–based approach, developed stochastic models capable of analytically characterizing the average service time of an application executed on a DCS [25, 28]. An important contribution made by the authors is that their approach considered heterogeneous CSs and the random information and task transfer-times. In addition, Dakhal *et al.* developed static and dynamic LB algorithm [27, 28, 67]. In [27], the stochastic model was extended to include reliability issues and two static LB algorithms were introduced. One algorithm is preemptively counteracting the

consequences of random failures while the other compensates for the occurrence failures upon the occurrence of them.

1.3 Contributions of this dissertation

The first major contribution of this dissertation is the development of a general probabilistic model for assessing performance and reliability in heterogeneous DCSs. A queueing theory setting has been used to construct the state-space of the stochastic process modeling the dynamics of the DCS. Next, the stochastic regeneration principle has been invoked to develop a recursive age-dependent analytical model for the random execution time taken by a DCS to concurrently serve an application. The model has been derived assuming a heterogeneous non-Markovian setting, that is, assuming that all the random events governing the system's dynamics follow arbitrarily specified probability distributions with heterogeneous parameters. Key in the development of the model is the inclusion of auxiliary age variables in the state vector. The auxiliary age variables are real-valued quantities that keep track of the memory of all the non-exponential random times, thereby relaxing the commonly made assumption about the Markovian dynamics of a DCS. This work presents also novel analytical approximations, as well as lower and upper bounds, for the random execution time of an application. These approximations reduce the dimension of the age-dependent state vector and yield a linear scalability in the number of CSs. The model for the random execution time has been employed to analytically characterize three metrics of practical applicability, namely, the average execution time of an application, quality-of-service (QoS) guarantees in executing an application, and the reliability in executing an application.

The second major contribution of this dissertation is the development of a novel model for spatially correlated failures in DCSs. To do so, the

topology of the DCS is abstracted as a graph, thereby capturing the geographical and logical correlations between the servers of the DCS. Next, patterns of failures exhibiting spatially correlated can be obtained by introducing a Markov random fields (MRFs) induced by the graph modeling the DCS. A practical contribution of this model is that the distribution function of failure patterns in the entire DCS can be obtained by specifying solely simple local interactions between neighboring servers. A key insight provided by the model for spatially correlated failures is that correlated failures effectively have more adverse effects on the reliability of DCSs than independent failures.

The third major contribution of this dissertation is the development of a class of DTR policies and an algorithm for optimizing the performance and reliability metrics characterized using the age-dependent regeneration theory. The simultaneous analysis and optimization of the performance and reliability conducted here have shed insights on the existence of a fundamental trade-off between minimizing execution time and maximizing the service reliability. In addition, the analysis of these metrics in a general setting has provide insights on the effect of network-delays on the accuracy of Markovian models. Results indicate that when the network delays are relatively large compared to service times, the error in estimating any of these metrics, as a result of falsely assuming exponentially distributed random delays, becomes significant, thereby necessitating the use of the age-dependent model developed in this work.

The fourth major contribution of this dissertation is the integration of the general model for a DCS and the model for correlated failures to yield a general framework for predicting and improving performance and reliability in DCSs in the presence of spatially correlated failures. To the best of the author's knowledge such a framework is not available in the literature. This framework is general and allows the assessment of performance and reliability in

non-Markovian DCSs where network anomalies may exhibit or not spatial correlation. Additionally, general DTR policies for improving performance and reliability in the face of spatially correlated anomalies is presented.

To date, the work developed in this dissertation has resulted in three journal papers [67–69], one book chapter [45], and three conference papers [30, 66, 70].

1.4 Dissertation overview

In Chapter 2 the problem of characterizing performance and reliability of applications executed on a DCS is stated. The mathematical definitions as well as the general assumptions made in this work are presented. In addition, the Markovian regeneration-based theory developed by Dhakal*et al.* in [25, 28] is reviewed.

In Chapter 3, a novel and rigorous analytical approach for characterizing performance and reliability metrics associated with the execution of parallel applications on a DCS is presented. Characterizations have been derived in a general setting where all the random times governing the dynamics of the DCS are assumed to follow arbitrarily specified probability distributions with heterogeneous parameters. In order to account for the efficient execution of applications, the modeling framework includes also the execution of a DTR policy by the system servers. The scalability problems associated with the characterizations is studied, and analytical approximations exhibiting a linear scalability, in the number of servers, are derived. In addition, analytical lower and upper bounds are also presented.

In Chapter 4, the problem of devising optimal DTR strategies for improving performance and reliability in DCS is formulated. A class of DTR policies executed in a synchronous and distributed manner by the servers in the DCS is presented, and an algorithm for calculating the optimal DTR policies is provided. The performance

and reliability are studied by means of theoretical predictions, MC simulations and experimental results conducted on a testbed DCS.

In Chapter 5, a stochastic model for spatially correlated failures on DCSs is presented. The developed model generates samples of correlated network anomalies by capturing the spatial geographical and logical interactions induced by the network topology of a DCS. Graph theory and MRFs theory are exploited to introduce "local specifications" of failures that, in turn, induce a global distribution function of failure patterns to all the servers in the DCS.

In Chapter 6, the general framework for predicting and improving performance and reliability in DCSs is combined with the model for spatially correlated failures. The combination of these two models allows the assessment of reliability in a more general framework where network anomalies may exhibit or not spatial correlation. In addition, a policy for improving reliability in the presence of spatially correlated anomalies is provided. Finally, Chapter 7 presents possible new lines of research for this work in the future.

Chapter 2

Markovian model for the execution time of parallel applications

In this chapter, the problem of characterizing performance and reliability of applications executed on a DCS is stated. The mathematical definitions as well as the general assumptions made in this work are presented. For completeness, the Markovian regeneration-based theory developed by Dhakal in [25] is reviewed.

2.1 Problem statement

Consider the problem of processing a parallel application on an n-server DCS, whose servers execute a synchronous DTR action at a given prescribed time. Suppose that the application belongs to the class of parallel applications with no data-dependence constraints between operations. Consequently, such application can be partitioned into an integer number, M, of indivisible and independent tasks. Suppose also that, at $t = 0$, an off-line scheduler has allocated m_j tasks in the queue of the jth server, where m_j is a non-negative integer and $M = \sum_{j=1}^{n} m_j$.

Assume that the service time of a task is random and depends only on the random service time of the server executing the task. This kind of heterogeneity in the task service time is referred to as processor-consistent heterogeneity [76,77]. Also, assume that computing servers can fail permanently at any random instant, and suppose also that no mechanism is provided by the DCS to totally or partially recover tasks from a failed server. Consequently, the application being executed on the system cannot be completed upon the failure of a server with unfinished tasks. This failure model is referred to as the crash-stop model [33, 63].

It has been established here that servers perform a DTR action at a prescribed time. The goal of this action is to efficiently execute the application in a parallel fashion. Let t_b denote the time at which the servers synchronously perform the DTR action. Also, let $l_{ij}(t_b) \equiv l_{ij}$ denote the number of tasks reallocated from the ith to the jth server at time t_b. By arranging the l_{ij} quantities in matrix form, the DTR policy can be denoted as $\mathbf{L} = (l_{ij})_{n\times n}$. In addition, it has been supposed here that upon failure a server broadcasts over the network a small fixed-sized failure-notice (FN) message in an attempt to inform the working servers about its faulty state. Due to the communication networks utilized in a DCS impose practical and unavoidable limitations, it has been assumed here that the exchange of either tasks or FN packets among any pair of servers faces a random communication delay. In particular, it has been assumed that due to the small size of the FN packets, their end-to-end delays depend solely on the heterogeneous end-to-end propagation time of each communication link. Also, it has been assumed also the mean transfer time of a group of l_{ij} tasks from the ith to the jth server follows the first-order approximation:

$$\bar{Z}_{ij} = a_{ij} l_{ij} + b_{ij}, \tag{2.1}$$

where a_{ij} and b_{ij} are positive constants (in seconds per task and seconds, respectively) that depend upon the communication channel connecting the ith and the jth server.

This first-order approximation captures the dependence of the mean transfer time on: (i) the number of tasks to be transferred; (ii) the end-to-end transmission time per task, through the parameter a_{ij} that is related to the bandwidth; and (iii) the combined effects of the absolute minimum end-to-end propagation time and arbitrary delays resulting from queueing (due to congestion), which can be represented by a single parameter, b_{ij}. Finally, it is further assumed that servers employ a reliable message-passing protocol. With this, tasks cannot be discarded in the network in situations like the failure of a server while transferring tasks to another server.

In the setting considered the time taken by the DCS to process the application is a random variable. Modeling this random variable is a extremely complicated because the execution time depends on both the DTR policy executed by the servers and on the interplay between all the aforementioned random times. For mathematical tractability, assumptions A1 and A2 are imposed on the random times driving the events occurring in the DCS:

Assumption A1. For any $j \neq k$, the following times are regarded as random and their pdfs are known: (i) W_{ki}: the service time of the ith task at the kth server, with pdf $f_{W_{ki}}(x)$; (ii) Y_k: the failure time of the kth server, with pdf $f_{Y_k}(x)$; (iii) X_{jk}: the transfer time of a FN message sent from the jth to the kth server, with pdf $f_{X_{jk}}(x)$; and (iv) Z_{ik}: the transfer time of l_{ik} tasks from the ith to the kth server, with pdf $f_{Z_{ik}}(x)$.

Assumption A2. All the random times listed in Assumption A1 are mutually independent.

In the next section the random execution time of an application as well as the three metrics regarded in this dissertation are briefly introduced. Mathematical definitions of these quantities will be presented in Section 2.2.1 after the construction of the Markovian state-space representation for the dynamics of the DCS. Later, in Sec-

tion 3.1.2 the Markovian assumption will be relaxed and a general (non-Markovian) state-space representation will be constructed. This new state-space will enable the recursive characterization of the performance and reliability metrics regarded in this dissertation.

2.2 Markovian model for the random execution time

In [25, 28], Dhakal*et al.* introduced a state-space model for the random execution time of applications in situations where servers may fail permanently or may fail and recover after a finite time. The state-space model was derived including an extra condition to Assumption A1. This extra condition stated that the random times W_{ki}, Y_k, X_{jk}, and Z_{ik} follow exponential distributions with parameters λ_{d_k}, λ_{f_k}, λ_{jk}, and $\tilde{\lambda}_{ik}$, respectively. When all the random events governing the dynamics of the underlying stochastic process follow exponential distributions, the model is commonly referred to as a Markovian model [11, 38].

Under this Markovian setting, Dhakal derived an analytical recursive characterization for both the average execution time of applications in an n-server DCS and the service reliability in a two-server DCS. For completeness, the characterization for the average completion time developed by Dhakal is presented here. In addition, the two-server characterization for the service reliability developed by Dhakal has been generalized here to the case of an n-server DCS. In the remaining of this section, material from [25, 28] will be freely drawn.

2.2.1 Markovian state-space model for DCSs

When the random events are assumed to follow exponential distributions, the memoryless property of such distribution guarantees the Markovian nature of the underlying stochastic process from which the execution time of an application is derived. Consequently, in a Markovian setting, the state-space $\mathbf{S}(t)$ of the continuous-time stochastic process $\{\mathbf{S}(t), t \geq 0\}$ is unambiguously specified by means of one state vector and two state matrices. These three objects describe the following discrete quantities: (i) the number of tasks queued at each server; (ii) the functional or dysfunctional state of each server in the system; and (iii) the amount of tasks in transit over the communication network.

Let $m_i(t)$ denote the queue length of the ith server in the DCS at time t, where $m_i(t)$ is a non-negative integer for all t and $i \in \{1, \ldots, n\}$. By stacking all these terms, the n-dimensional vector $\mathbf{m}(t)$, termed as *system-queue state vector*, is introduced and its ith element is precisely $m_i(t)$. In particular, at $t = 0$, the system-queue state vector is $\mathbf{m}(0) = \begin{pmatrix} m_1 & m_2 \ldots m_n \end{pmatrix}^T$ due to the assumption on an off-line scheduler allocating m_i tasks in the queue of the ith server at $t = 0$.

Let $f_i(t)$ be a binary variable representing the working ("1") or failed ("0") state of the ith server at time t. For $i \neq j$, let $f_{ij}(t) = 1$ (correspondingly, $f_{ij}(t) = 0$) indicate that the jth server is functioning (correspondingly, faulty) as perceived by the ith server at time t. By arranging all these variables in an n-by-n matrix, the *system-function state matrix* $\mathbf{F}(t)$ can be introduced. Note that the random transfer time of FN packets introduce uncertainty on the functioning state that a server perceives about the other servers in the DCS. For example, for $n = 3$ and $t = t_0$ the configuration $\mathbf{F}(t_0) = \left(\begin{smallmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{smallmatrix}\right)$ corresponds to a state where server 2 has failed and only server 3 is aware of the failed state of server 2.

Additionally, let $c_{ij}(t)$ be a non-negative integer denoting the number of tasks

being transferred over the network from the ith to the jth server at time t. As in the case of the system-function state matrix, by arranging the $c_{ij}(t)$ terms in matrix form the non-negative integer-valued matrix $\mathbf{C}(t)$ can be obtained. Such matrix has been termed as the *network state matrix* and specifies all the tasks being transferred over the network at time t.

Next, Dhakal *et al.* defined in [25,28] the state-space representation for an n-server DCS as the concatenation of the system-queue state vector, system-function state matrix and the network state matrix, i.e., $\mathbf{S} = (\mathbf{m}, \mathbf{F}, \mathbf{C})$. In this dissertation, the definition of the state-space for a DCS provided by Dhakal is employed to formally introduce the stochastic process $\{\mathbf{S}(t), t \geq 0\}$, which characterizes the stochastic dynamics of the DCS in a Markovian setting. Note that the vector $\mathbf{m}$ and the matrices $\mathbf{F}$ and $\mathbf{C}$ have finite dimensions and take values on the finite discrete sets $\Omega_1 = \{0, 1, \ldots, M\}^n$, $\Omega_2 = \{0, 1\}^{n^2}$ and $\Omega_3 = \{0, M\}^{n^2}$, respectively. For $\Omega = \Omega_1 \times \Omega_2 \times \Omega_3$, we can always define any one-to-one mapping $h : \Omega \to \mathcal{I}$ such that, for each possible value of the concatenated matrix $(\mathbf{m}, \mathbf{F}, \mathbf{C})$ in Ω, $h(\mathbf{m}, \mathbf{F}, \mathbf{C})$ assigns a positive integer in the index set $\mathcal{I} = \{1, 2, \ldots, \kappa\}$, where κ is the cardinality of Ω.

2.3 Mathematical definition of performance and reliability metrics

Let $\{\mathbf{S}(t), t \geq 0\}$ be a continuous-time stochastic process, with state-space $\mathbf{S}(t)$, modeling the queue-length of the servers, the number of tasks queued in the network and the failed or working state of the servers. Suppose that at time $t = 0$, the configuration of the state-space of the DCS is $\mathbf{S}_0 = \mathbf{S}(0) = (\mathbf{m}_0, \mathbf{F}_0, \mathbf{C}_0)$. Moreover, exploiting the definition of the indexing mapping $h(\cdot)$, we label the initial configuration for the DCS as $\ell_0 = h(\mathbf{S}_0)$.

Definition 1. The *random execution time of an application* is defined as the random time taken by the DCS to serve an application if servers execute the DTR policy $\mathbf{L}$ at time $t = t_b$ and the initial system configuration is as specified by $\mathbf{S}_0$. Mathematicaly:

$$T_{\ell_0}(t_b) \triangleq \inf\{t > 0 : \mathbf{m}(t) = \mathbf{0} \text{ and } \mathbf{C}(t) = \mathbf{0}\}. \tag{2.2}$$

The random execution time of an application is denoted as $T(t_b, \mathbf{L}; \mathbf{S}_0)$ and will be mathematically defined from the stochastic process in (2.2). It must be noted that since servers can fail permanently with non-zero probability, the execution time is defined to be infinite when at least one task remains queued at a server that has already failed. Note also that in the special case where servers are completely reliable, the application execution time is always finite. From this definition for the random execution time, the following performance and reliability metrics can be introduced.

Definition 2. The *average execution time of an application*, denoted as $\overline{T}_{\ell_0}(t_b)$, is the performance metric defined as the expected value of the random application execution time, that is:

$$\overline{T}_{\ell_0}(t_b) \triangleq \mathsf{E}[T_{\ell_0}(t_b)]. \tag{2.3}$$

The average execution time is critical to assess the speed-up in the execution of applications when executed in parallel on a DCS. The average execution time is a reasonable metric (i.e., it takes a finite value) only in settings where servers are completely reliable or in settings where servers are allowed to recover after a failure. In this dissertation, the average execution time is defined for the case where servers are completely reliable, i.e., when Y_k is equal to infinity almost surely for all k.

Definition 3. The *QoS in executing an application*, denoted as $Q_{\ell_0}(t_b, T_M)$, is a performance metric defined as probability that the application can be entirely executed by the user-specified due time T_M, that is:

$$Q_{\ell_0}(t_b, T_M) \triangleq \mathsf{P}\{T_{\ell_0}(t_b) < T_M\}. \tag{2.4}$$

The QoS is a reasonable metric in settings where server nodes may or may not fail. The QoS is a metric of interest to system users and analysts, specially in real-time or in time constrained applications.

Definition 4. The *service reliability*, denoted as $R_{\ell_0}(t_b)$, is a metric defined as the probability that the application can be entirely executed by the system, that is:

$$R_{\ell_0}(t_b) \triangleq \mathsf{P}\{T_{\ell_0}(t_b) < \infty\}. \tag{2.5}$$

The service reliability is an important metric for assessing the dependability of applications executed on DCSs that do not tolerate down times. The metric is a reasonable only when servers can fail without recovery (crash-stop model) and/or in settings where applications cannot continue their execution after a failure. It must be noted that the service reliability is a special case of the QoS for which the due time to execute the application is finite.

2.3.1 Markovian characterization of the performance and reliability metrics

With all these definitions at hand, Dhakal *et al.* developed a regeneration theory for recursively characterizing the average execution time of a n-server DCS [25,28]. The characterization consists in a system of coupled difference-differential equations in the time variable ξ, which represents any arbitrary task reallocation instant. The system of coupled difference-differential equations are stated in Theorem 1 and the initial condition for $\xi = 0$ is presented in Theorem 2.

Theorem 1 (Dhakal [25], Dhakal *et al.* [28])**.** *Consider an n-server DCS satisfying Assumptions A1 and A2. Suppose also that all the random times listed in A1 follow exponential distributions, and that all the servers perform a synchronous DTR action at the time* $\xi \geq 0$*. For any* $\ell \in \mathcal{I}$*, the average execution time of an application*

satisfies the set of coupled, difference-differential equation:

$$\frac{d}{d\xi}\overline{T}_{\ell}(\xi) = \sum_{i=1}^{n} \lambda_{d_i} \overline{T}_{\ell_i}(\xi) - \sum_{i=1}^{n} \lambda_{d_i} \overline{T}_{\ell}(\xi) + 1, \tag{2.6}$$

where $\mathbf{S} = (\mathbf{m}, \mathbf{F}, \mathbf{0})$ *denotes an arbitrary initial system configuration,* $\ell = h(\mathbf{m}, \mathbf{F}, \mathbf{0})$*,* $\ell_i = h(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{F}, \mathbf{0})$*,* $\boldsymbol{\delta}_i$ *denotes an* n*-dimensional vector with all its entries equal to zero except that its* i*th element is equal to one, and* $m_i - 1$ *is set to zero when* $m_i = 0$*.*

Proof: See [25] Section 3.2 and Dhakal *et al.* [28] Appendix B.

Theorem 2 (Dhakal [25], Dhakal *et al.* [28])**.** *Consider an* n*-server DCS satisfying Assumptions A1 and A2. Suppose also that all the random times listed in A1 follow exponential distributions. For any* $\ell \in \mathcal{I}$*, the initial condition* $\overline{T}_{\ell}(0)$ *associated with the average execution time of an application satisfies the general algebraic recursion:*

$$\overline{T}_{\ell}(0) = \sum_{i=1}^{n} \frac{\lambda_{d_i}}{\lambda} \overline{T}_{\ell_i}(0) + \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \frac{\tilde{\lambda}_{ji}}{\lambda} \overline{T}_{\ell'_{ji}}(0) + \frac{1}{\lambda}, \tag{2.7}$$

where $\mathbf{S} = (\mathbf{m}, \mathbf{F}, \mathbf{C})$ *denotes an arbitrary initial system configuration,* $\ell = h(\mathbf{m}, \mathbf{F}, \mathbf{C})$*,* $\ell_i = h(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{F}, \mathbf{C})$*,* $\ell'_{ji} = h(\mathbf{m} + l_{ji}\boldsymbol{\delta}_i, \mathbf{F}, \mathbf{C}^{(ji)})$*, and* $\lambda = \sum_{i=1}^{n} \lambda_{d_i} + \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \tilde{\lambda}_{ij}$*.*

Proof: See [25] Section 3.2.

It must be noted here that (2.6) and (2.7) characterize the rate of change in the average service time as a function of the reallocation action and the DTR policy for any initial task allocation and for any network state. To calculate the average execution time, a system of difference-differential equations must be constructed from (2.6). This system of equations involves recursions in terms of the state vector $\mathbf{S}$. Therefore, the configuration of the DCS at $t = 0$ must be specified. This configuration is $\mathbf{S}_0 = (\mathbf{m}_0, \mathbf{F}_0, \mathbf{0})$, where $\mathbf{m}_0 = (m_1\ m_2 \ldots m_n)^T$ represents the initial allocation

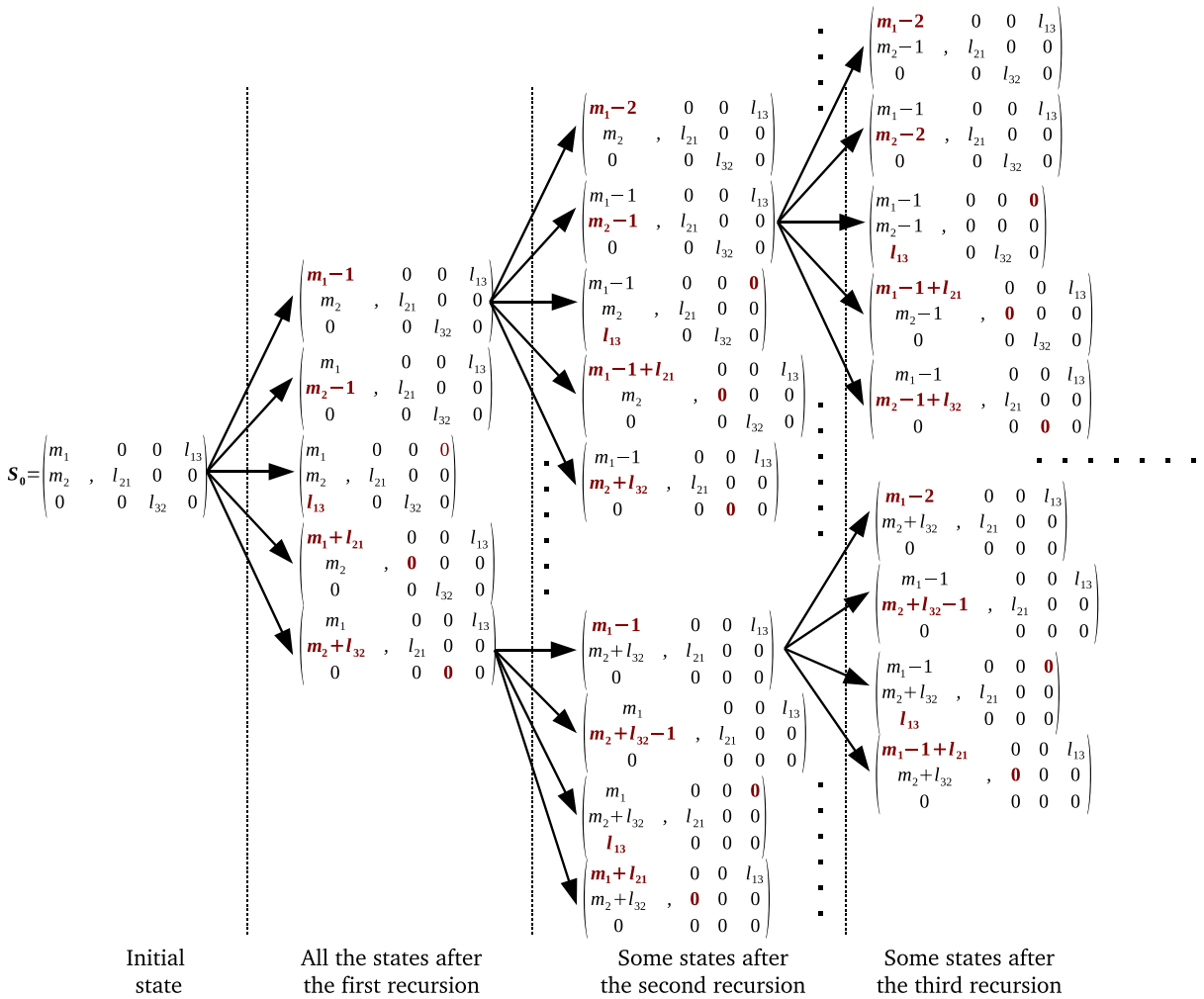


Figure 2.1: Example of the recursions generated from a generic initial state $\mathbf{S}_0$ in the case of a non-Markovian DCS.

of tasks, $\mathbf{F}_0 = \mathbf{1}$ is an all-ones matrix representing that all the servers are functioning and $\mathbf{C}_0 = \mathbf{0}$ is the null-matrix representing that no-task is being transferred in the network. Figure 2.1 shows an example of the recursions generated starting form the initial state $\mathbf{S}_0$. The transition of the states occurs from left to right. Only a single transition is allowed and those quantities that have changed are colored and written in bold font. Note that by labelling the states depicted in fig. 2.1 a particular form for the mapping $h(\mathbf{m}, \mathbf{F}, \mathbf{C})$ can be defined. Also, it must be noted that the number

of states created after a couple of recursions grows exponentially.

Similarly, it has been proven by the author of this dissertation in [67] that the service reliability of an n-server DCS satisfies the difference-differential equation in Theorem 3, which has an initial condition for $t_b = 0$ given in Theorem 3.

Theorem 3 (Dhakal [25], Pezoa *et al.* [67])**.** *Consider an n-server DCS satisfying Assumptions A1 and A2. Suppose that all the random times listed in A1 follow exponential distributions, and that all the servers perform a synchronous DTR action at the time $\xi \geq 0$. For any $\ell \in \mathcal{I}$, the service reliability satisfies the set of coupled difference-differential equations:*

$$\frac{d}{d\xi}R_{\ell}\big(t_b\big) = \sum_{i=1}^{n}\lambda_{d_i}R_{\ell_i}\big(t_b\big) + \sum_{i=1}^{n}\lambda_{f_i}R_{\ell_i'}\big(t_b\big) + \sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\lambda_{ij}R_{\ell_{ij}}\big(t_b\big) - \lambda R_{\ell}\big(t_b\big), \qquad (2.8)$$

where $\mathbf{S} = (\mathbf{m}, \mathbf{F}, \mathbf{0})$ denotes an arbitrary initial system configuration, $\ell = h(\mathbf{m}, \mathbf{F}, \mathbf{0})$, $\ell_i = h(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{F}, \mathbf{0})$, $\ell_i' = h(\mathbf{m}, \mathbf{F}^{(ii)}, \mathbf{0})$, $\ell_{ij} = h(\mathbf{m}, \mathbf{F}^{(ji)}, \mathbf{0})$, $\lambda = \sum_{i=1}^{n}\big(\lambda_{d_i} + \lambda_{f_i}\big) + \sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\lambda_{ij}$ and the the matrix $\mathbf{A}^{(ij)}$ is identical to the matrix $\mathbf{A}$ but with its ijth component set to zero.

Proof: See Pezoa *et al.* [67] Appendix.

Theorem 4 (Dhakal [25], Pezoa *et al.* [66])**.** *Consider an n-server DCS satisfying Assumptions A1 and A2. Suppose also that all the random times listed in A1 follow exponential distributions. For any $\ell \in \mathcal{I}$, the initial condition $R_{\ell}\big(t_b\big)$ associated with the service reliability of an application satisfies the general algebraic recursion:*

$$\begin{aligned} R_{\ell}\big(0\big) = &\sum_{i=1}^{n}\frac{\lambda_{d_i}}{\lambda}R_{\ell_i}\big(0\big) + \sum_{i=1}^{n}\frac{\lambda_{f_i}}{\lambda}R_{\ell_i'}\big(0\big) + \sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\frac{\lambda_{ij}}{\lambda}R_{\ell_{ij}}\big(0\big) \\ &+ \sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\frac{\tilde{\lambda}_{ij}}{\lambda}R_{\ell_{ij}'}\big(0\big), \end{aligned} \qquad (2.9)$$

where $\mathbf{S} = (\mathbf{m}, \mathbf{F}, \mathbf{C})$ *denotes an arbitrary initial system configuration,* $\ell = h(\mathbf{m}, \mathbf{F}, \mathbf{C})$*,* $\ell_i = h(\mathbf{m}-\boldsymbol{\delta}_i, \mathbf{F}, \mathbf{C})$*,* $\ell'_i = h(\mathbf{m}, \mathbf{F}^{(ii)}, \mathbf{C})$*,* $\ell_{ij} = h(\mathbf{m}, \mathbf{F}^{(ji)}, \mathbf{C})$*,* $\ell'_{ij} = h(\mathbf{m}+l_{ji}\boldsymbol{\delta}_i, \mathbf{F}, \mathbf{C}^{(ji)})$*, and* $\lambda = \sum_{i=1}^{n} \left(\lambda_{d_i} + \lambda_{f_i}\right) + \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} (\lambda_{ij} + \tilde{\lambda}_{ij})$.

Proof: See Pezoa *et al.* [66] Appendix.

2.4 Service reliability in a Markovian setting

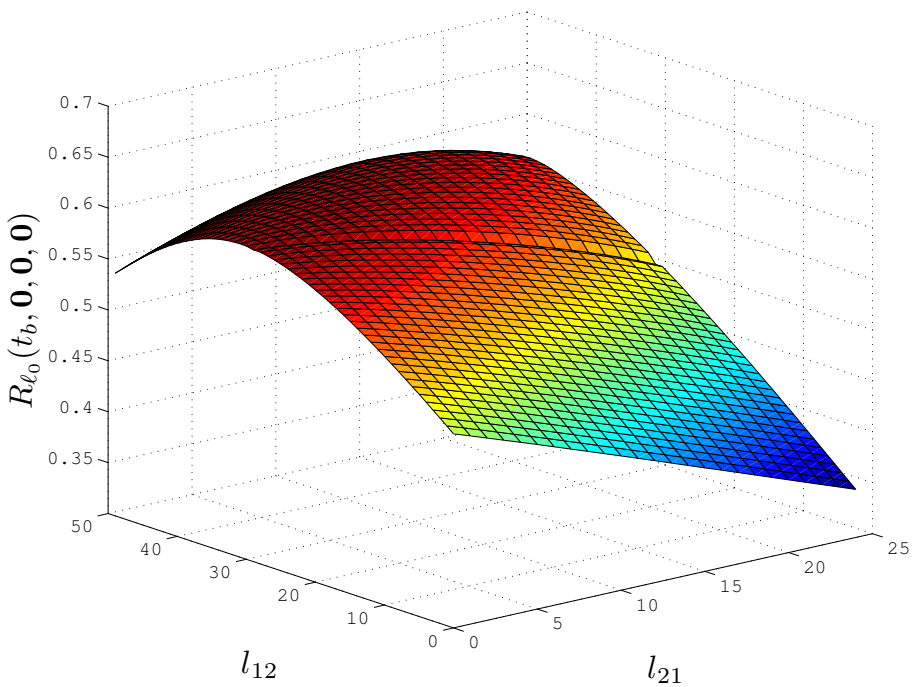


Figure 2.2: Service reliability of a two-server DCS as a function of the number of tasks exchanged among the servers.

The service reliability of a two-server DCS has been assessed exploiting the analytical characterization (2.9). The initial allocation of tasks onto the servers is $m_1 = 50$ tasks and $m_2 = 25$ tasks, while the average failure times of the servers are

$\lambda_{f_1}^{-1}$=300 s and $\lambda_{f_2}^{-1} = 100$ s. The service rates of each server are $\lambda_{d_1} = 0.1682$ tasks per second (tps) and $\lambda_{d_2} = 0.4978$ tps. The mean arrival times of each FN packet are $\lambda_{12}^{-1} = 4.6402$ s and $\lambda_{21}^{-1} = 1.6659$ s. Finally, the parameters of the first order approximation for the average transfer time per task are: $a_{12} = 0.243$, $a_{21} = 0.339$ (in seconds per task) and $b_{12} = 1.971$, $b_{21} = 1.652$ (in s). It must be commented that the parameters of the DCS have been estimated from data obtained after conducting training experiments on a two-server testbed as reported in [66].

The estimated service reliability of the two-server DCS is calculated by solving the system of recurrence equations generated by (2.9), for the initial state $\mathbf{m}_0 = (m_1\ m_2)'$, $\mathbf{F} = \mathbf{1}$ and $\mathbf{C} = \left(\begin{smallmatrix} 0 & l_{21} \\ l_{12} & 0 \end{smallmatrix}\right)$. Figure 2.2 depicts the service reliability as a function of both the number of tasks exchanged from server 1 to server 2, l_{12}, and the number of tasks exchanged from server 2 to server 1, l_{21}. Note that an improper selection of the amount of tasks to migrate between the servers can produce a significant reduction on the service reliability, as is depicted in the case of choosing $l_{12} = 0$ tasks and $l_{21} = 25$ tasks. This reduction is a consequence of the following situation: for small values of l_{12}, server 1 (which is the slowest server) keeps most of its initial load. Consequently, the load distribution is unbalanced between the server even after the DTR is executed. Therefore, the time required to serve the application increases and, as a consequence, the service reliability is reduced. When l_{12} approaches to 50, the first server transfers most of its initial load to the second server. Hence, almost all the tasks are queued, first in the network and later at server 2. Since server 2 is the most unreliable server, it is expected to observe a reduction in the service reliability because of the queuing of tasks and the unwise decision of transferring most of the tasks to the less reliable server. Finally, it can be noticed that the optimal DTR policy corresponds to $l_{12}^* = 30$ tasks and $l_{21}^* = 0$ tasks, which provides a theoretical optimal service reliability of 0.601.

2.5 On the validity of application of the Markovian models

Assuming that all the random events governing the dynamics of a DCS follow exponential distributions is a popular yet idealized assumption. The memoryless property of the exponential distribution guarantees that the underlying stochastic process is a continuous-time Markov chain. Even though the analysis and modeling problem is still hard in this simplified setting, some relief can be obtained in the analysis because of the existence of a large amount of tools from Markov chain theory that can be used to analyze the dynamics of the DCS [11, 38]. However, several researchers have raised questions about the accuracy of the exponential distribution in modeling real phenomena.

For example, in [67] the accuracy of the Markovian regenerative model to predict the service reliability of a testbed DCS was evaluated by means of MC simulations. Using experimental data collected from the testbed, Pareto probability distributions were fitted for the actual transfer and service times. In addition, from the sampled mean of the data, exponential distributions were also fitted to approximate the results by means of the Markovian theory developed. The Pareto models were evaluated via MC simulations, and the service reliability was estimated with a 95% confidence. It was observed that the Markovian model for reliability was very accurate and yielded a relative approximation error below 4%. However, further simulations showed that, as the ratio between the average transfer time and the average service time of the servers increases, the exponential approximation looses its accuracy in predicting the reliability. Specifically, approximation errors of 120% were found when the ratio between the times was five. This observation raised the following research questions: under which conditions the Markovian models yield "good approximations" for the service reliability? How accurate is the Markovian model for the average execution

time? Are the amount error in the approximation for the reliability of the same order of magnitude as in the case of the average execution time? Does the modeling error affect the DTR policies? In the upcoming chapters of this dissertation these question will be answered.

Chapter 3

Non-Markovian model for the execution time of parallel applications

In this chapter, a novel age-dependent analytical theory is derived for characterizing three performance and reliability metrics of great interest to system analysts and designers. These characterizations have been derived in a general setting where all the random times governing the dynamics of the DCS are assumed to follow arbitrary probability distributions with heterogeneous parameters. In particular, the Markovian models derived by Dhakal *et al.* in [25, 28] are special cases of the general characterizations presented here. The scalability, in the number of servers, of the characterizations is studied, and analytical approximations exhibiting a linear scalability are derived. In addition, analytical lower and upper bounds have been also derived and presented.

3.1 Age-dependent characterization of the random execution time

The discrete state-space model as well as the recursive characterizations for the average execution time and the service reliability presented in [28, 67], and reviewed in Chapter 2, were derived under Assumption A1, A2 and the extra condition that all the random times follow exponential distributions with known parameters. When the distributions of the random events follow any arbitrary probability distribution, memory is introduced into the system dynamics and destroying the Markovian property of the underlying stochastic process. In order to retain the Markovian property of the process the memory of the non-exponential distributions must be tracked. To do so, in this dissertation real-valued age variables have been introduced in the analysis to supplement the discrete state-space model presented in the previous chapter. The inclusion of these age variables captures the memory of the non-Markovian random times and also enables the construction of a hybrid discrete and continuous state-space representation for the underlying stochastic process governing dynamics of the DCS.

3.1.1 Auxiliary age variables

Auxiliary real-valued age variables have been employed in this dissertation for analyzing the stochastic dynamics of a DCS. To illustrate how the concept of age variables has been exploited here, suppose that T is random variable representing some random time. Loosely speaking, if it is known that a units of time have elapsed for the random time T, then the remaining random time $T_a = T - a$ can be introduced as the replacement of the random time T measured from a, and also, one can think of T_a as the aged version of T with age a. Proceeding formally, the age parameter,

a, associated with the random time T, is defined as the non-negative, real-valued quantity that defines on the event $A = \{T \geq a\}$, the random time $T_a = T - a$. The random time T_a is the aged version of T with age parameter a, and has a pdf equal to the conditional pdf of T given that A has occurred, that is, $f_{T_a}(t; a) = f_{T|A}(t|a)$. It must be noted that if T is exponentially distributed, then the pdfs of T and T_a are identical due to the memoryless property of the exponential distribution.

In this dissertation the relationship between a random variable and its aged version has been exploited as follows: As soon as a random time T is triggered by some event, its associated age variable is set to zero, and as time elapses, the age variable keeps track of the age of T and adjusts accordingly the pdf of T to show the effect of the elapsed time on its likelihood. If a random time has not been triggered by an event, the age variable associated with it is set to infinity.

It must be commented that in [20] Cox introduced the method of supplementary variables for analyzing a general class of continuous-time stochastic process with discrete state spaces. In his work, Cox stated that if at any instant the discrete state variables as well as a finite number of continuous variables are specified, a non-Markovian process can be converted into a Markovian one. His method has been employed in the form of expended life-times, which record the elapsed time associated with a random variable and can be though of as increasing timers [38], or in the form of rest variables, which record the remaining time associated with a random variable and can be though of as decreasing timers [38]. Even though the expended life-time variables are conceptually the same as the auxiliary age variables employed in this dissertation, this represents by no means a direct application of the method introduced by Cox. Here, the stochastic process driving the dynamics of the DCS has been constructed from basic principles and a stochastic regeneration theory has been developed to rigorously prove the regenerative behavior of the process, while in [20] Cox developed a general example for a classical G/G/n queueing system.

3.1.2 Age-dependent state-space model for DCSs

In a non-Markovian setting, the state-space model for an n-server DCS is partially described by the system-queue state vector, the system-function state matrix and the network state matrix. The configuration of a non-Markovian, n-server DCS is completely and unambiguously specified if $\mathbf{m}(t)$, $\mathbf{F}(t)$, and $\mathbf{C}(t)$ are supplemented with one more vector, which is associated with $\mathbf{m}(t)$, and two extra matrices, which are associated with $\mathbf{F}(t)$ and $\mathbf{C}(t)$.

Let a_{M_i} and a_{F_i} be age variables associated with the random service time of a task at the ith server and the random failure time of the ith server, respectively, with $i = 1, \ldots, n$. Also, let $a_{F_{ij}}$ be the age variable connected to the random transfer time of a FN packet from the ith to the jth server, with $i, j = 1, \ldots, n$, $i \neq j$. All these age variables can be arranged in the column vector $\mathbf{a}_M$ and the n-by-n matrix $\mathbf{a}_F$. The $\mathbf{a}_M$ vector contains the a_{M_i} age variables and the $\mathbf{a}_F$ matrix contains both the a_{F_i} variables (at the diagonal of the matrix) and the $a_{F_{ij}}$ variables (at the off-diagonal positions). Similarly, let $a_{C_{ik}}$ be the age variable associated with the random transfer of l_{ik} tasks from the ith to the kth server. These age variables can also be arranged in matrix form to obtain $\mathbf{a}_C$, whose ikth component is $a_{C_{ik}}$.

At this point, the *system-age state matrix* can be defined as the concatenated matrix $\mathbf{a} \triangleq (\mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$. Further, for a given time t we define the *age-dependent system-state matrix* as the concatenated matrix $\mathbf{S}(t) \triangleq \big(\mathbf{m}(t), \mathbf{F}(t), \mathbf{C}(t), \mathbf{a}(t)\big)$, which describes completely the state of a non-Markovian n-server DCS. It must be noted again that in a Markovian setting the memoryless property of the exponential distribution makes the system-age matrix unnecessary; therefore, the system-state matrix reduces to the age independent case $\mathbf{S}(t) = \big(\mathbf{m}(t), \mathbf{F}(t), \mathbf{C}(t)\big)$.

Now, equipped with the age-dependent state vector, the stochastic process can be extended to the non-Markovian case. Let $\{\mathbf{S}(t), t \geq 0\}$ be the continuous-time

stochastic process, with hybrid continuous and discrete state-space $\mathbf{S}(t)$, that characterizes the dynamics of the DCS in a general setting where arbitrary probability distribution govern the random events of the system. Recall that we can always define any one-to-one mapping $h : \Omega \to \mathcal{I}$ such that, for each possible value of the discrete part of the age-dependent state-vector, i.e., $(\mathbf{m}, \mathbf{F}, \mathbf{C})$, $h(\mathbf{m}, \mathbf{F}, \mathbf{C})$ assigns a positive integer in the index set $\mathcal{I} = \{1, 2, \ldots, \kappa\}$, where κ is the cardinality of Ω.

Definition 5. The *age-dependent random execution time of the application* can be mathematically defined from the generalized stochastic process as

$$T_{\ell_0}(t_b, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) \triangleq \inf\{t > 0 : \mathbf{m}(t) = \mathbf{0} \text{ and } \mathbf{C}(t) = \mathbf{0}\}, \tag{3.1}$$

with $\ell_0 = h(\mathbf{m}_0, \mathbf{F}_0, \mathbf{C}_0)$.

Note that the right-hand side of the definitions (2.2) and (3.1) are identical; however, in (3.1) we make a explicit reference to the age-dependent nature of the random execution time of an application.

Definition 6. The *age-dependent average execution time of an application*, denoted as $\overline{T}_{\ell_0}(t_b, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$, is the performance metric defined as the expected value of the age-dependent random application execution time, that is:

$$\overline{T}_{\ell_0}(t_b, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) \triangleq \mathsf{E}[T_{\ell_0}(t_b, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)]. \tag{3.2}$$

Definition 7. The *age-dependent QoS in executing an application*, denoted as $Q_{\ell_0}(t_b, T_M, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$, is a performance metric defined as probability that the application can be entirely executed by the user-specified due time T_M, that is:

$$Q_{\ell_0}(t_b, T_M, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) \triangleq \mathsf{P}\{T_{\ell_0}(t_b, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) < T_M\}. \tag{3.3}$$

Definition 8. The *age-dependent service reliability*, denoted as $R_{\ell_0}(t_b, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$, is a metric defined as the probability that the application can be entirely executed

by the system, that is:

$$R_{\ell_0}(t_b, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) \triangleq \mathsf{P}\{T_{\ell_0}(t_b, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) < \infty\}. \tag{3.4}$$

3.2 Age-dependent regeneration-based approach and recursive characterization of metrics

3.2.1 Age-dependent regeneration theory

The theory of stochastic regeneration has been exploited here to derive an age-dependent recursive characterization of the metrics (2.3)–(2.5). The key idea is to define an age-dependent regeneration event and analyze the stochastic process emerging immediately after the *first occurrence* of this event. The *age-dependent regeneration event* is defined as the first occurrence of either the service of a task at any server, the failure of any server, the reception of a FN packet by any server, or the reception of a group of tasks by any server. The point here is that upon the occurrence of the regeneration event, a *fresh* copy of the original stochastic process emerges at the regeneration time albeit with a *new* initial configuration that transpires from the regeneration event. Unlike the regeneration-based approach taken in [28, 67], the age-dependent regeneration theory presented here must employ the information supplemented by the age variables in order to yield a regenerative stochastic process.

Proceeding formally, consider the process $\{\mathbf{S}(t), t \geq 0\}$ and suppose that at time $t = t_0$ the system configuration is as specified by $\mathbf{S} = (\mathbf{m}, \mathbf{F}, \mathbf{C}, \mathbf{a})$. The *age-dependent regeneration time*, denoted by $\tau_{\mathbf{a}}$, is defined as the minimum of the following four random variables: the time to the *first* task service by any server, the time to the *first* occurrence of failure at any server, the time to the *first* arrival of a FN packet at any server, or the time to the *first* arrival of a group of tasks at any server. Given the

system-age $\mathbf{a}$, for $t \geq t_0$ the random times listed in Assumption A1 can be replaced by their aged versions, thereby the age-dependent regeneration time can be defined mathematically as:

$$\tau_{\mathbf{a}} \stackrel{\triangle}{=} \min\big(\min_{k} W_{k1}, \min_{k} Y_k, \min_{j \neq k} X_{jk}, \min_{k,i} Z_{ik}\big), \tag{3.5}$$

where the subscript $\mathbf{a}$ emphasizes the dependency of the regeneration time on all the age variables associated with the non-exponential random times. The upcoming example describes how the age-dependent regeneration time and the age-dependent system-state matrix can be used to yield a recursive characterization for the execution time of an application.

Suppose that the first event occurring in the DCS happens to be the execution of a task at the ith server at $t = s$, for $t_0 < t$. The occurrence of this event implies that all the random times governing the DCS have aged by s units of time (in addition to the ages specified in $\mathbf{a}$) and there is one less task queued at the ith server; all the other dynamics remain unchanged. Thus, the occurrence of the event $\{\tau_{\mathbf{a}} = s, \tau_{\mathbf{a}} = W_{i1} - a_{M_i}\}$ gives birth to a new DCS at $t = s$, represented by $\{\mathbf{S}(t), t \geq s\}$, that is statistically identical to the original process while having a new initial configuration $\mathbf{S}' = (\mathbf{m}' \mathbf{F}', \mathbf{C}', \mathbf{a}')$ resulting from the regeneration event $\{\tau_{\mathbf{a}} = s, \tau_{\mathbf{a}} = W_{i1} - a_{M_i}\}$. More precisely, the new initial system configuration is as follows: $\mathbf{m}'$ is identical to $\mathbf{m}$ but with one unit less at its ith element, $\mathbf{F}' = \mathbf{F}$, $\mathbf{C}' = \mathbf{C}$, and the new system-age matrix is $\mathbf{a}' = \mathbf{a} + s$ with the ith component of $\mathbf{a}'_M$ set to zero if at least one task remains queued at the ith server and set to infinity otherwise. Similar transformations on the initial configuration are observed when the regeneration event is any of the remaining events, namely, the failure of the ith server, the arrival of a FN packet from server j to server k, or the arrival of the ith group of tasks to the kth server.

3.2.2 Characterization of the performance metrics

Before stating Theorems 1 and 2, we introduce some useful definitions. Let us define the term $G_X(\alpha) \triangleq \mathsf{P}\{X = \tau_{\mathbf{a}} | \tau_{\mathbf{a}} = \alpha\} f_{\tau_{\mathbf{a}}}(\alpha)$, where X is any of the random times listed in Assumption 1, $f_{\tau_{\mathbf{a}}}(\alpha)$ is the pdf of the age-dependent regeneration time $\tau_{\mathbf{a}}$ and $\mathsf{P}\{X = \tau_{\mathbf{a}} | \tau_{\mathbf{a}} = \alpha\}$ is the probability that the regeneration event is $\{\tau_{\mathbf{a}} = X\}$ conditional on the event $\{\tau_{\mathbf{a}} = \alpha\}$. Both, the pdf of the the age-dependent regeneration time and the latter conditional probability can be computed explicitly, either analytically or numerically, using (A.10) and (A.11), respectively. Note that the vector $\mathbf{m}$ and the matrices $\mathbf{F}$ and $\mathbf{C}$ have finite dimensions and take values on the finite discrete sets $\Omega_1 = \{0, 1, \ldots, M\}^n$, $\Omega_2 = \{0, 1\}^{n^2}$ and $\Omega_3 = \{0, M\}^{n^2}$, respectively. For $\Omega = \Omega_1 \times \Omega_2 \times \Omega_3$, we can define any one-to-one mapping $h : \Omega \to \mathcal{I}$ such that, for each possible value of the concatenated matrix $(\mathbf{m}, \mathbf{F}, \mathbf{C})$ in Ω, $h(\mathbf{m}, \mathbf{F}, \mathbf{C})$ assigns a positive integer in the index set $\mathcal{I} = \{1, 2, \ldots, \kappa\}$, where κ is the cardinality of Ω. Finally, let $\mathbf{m} = (r_1, \ldots, r_n)^T$, $\mathbf{F} = (f_{ij})_{n \times n}$, $\mathbf{C} = (c_{ij})_{n \times n}$, $\mathbf{a}_M$, $\mathbf{a}_F$, and $\mathbf{a}_C$ denote an arbitrarily specified initial configuration for the DCS, where r_i is the number of tasks queued at the ith server, $f_{ij} \in \{0, 1\}$ for all i, j and $c_{ij} \in \{0, 1, \ldots, M\}$.

Theorem 5 (Age-dependent characterization for the average execution time)**.** *Consider an n-server DCS whose servers perform a synchronous DTR action at the time $\xi \geq 0$. For any $\ell \in \mathcal{I}$, the average application execution time satisfies the system of recursive, coupled integral equations in ξ:*

$$\overline{T}_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) = \int_0^\xi \sum_{i=1}^{n} G_{W_{i1}}(\alpha) \overline{T}_{\ell_i}\big(\xi - \alpha, (\mathbf{a}_M + \alpha)^{(i)}, \mathbf{a}_F + \alpha, \mathbf{a}_C + \alpha\big)\, d\alpha + \int_0^\xi \alpha f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha + \big(1 - F_{\tau_{\mathbf{a}}}(\xi)\big)\big(\xi + \overline{T}_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)\big), \quad (3.6)$$

where recursions are carried out in the discrete variables $\mathbf{m}$ and $\mathbf{F}$, the vectors $\mathbf{m}$ and $\mathbf{a}_M$ and the matrices $\mathbf{F}$, $\mathbf{0}$, $\mathbf{a}_F$, and $\mathbf{a}_C$ denote an arbitrarily specified initial system

configuration, $\ell = h(\mathbf{m}, \mathbf{F}, \mathbf{0})$*,* $\ell_i = h(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{F}, \mathbf{0})$*,* $\boldsymbol{\delta}_i$ *denotes an* n*-dimensional vector with all its entries equal to zero except that its* i*th element is equal to one, the vector* $\mathbf{v}^{(i)}$ *is identical to the vector* $\mathbf{v}$ *but with its* i*th component set to zero, and* $\overline{T}_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ *is the initial condition related to the* ℓ*th integral equation.*

Proof: See Appendix A.

Theorem 6 (Age-dependent characterization for the QoS)**.** *Consider an* n*-server DCS whose servers perform a synchronous DTR action at the time* $\xi \geq 0$*. For any* $\ell \in \mathcal{I}$*, the QoS in executing an application by a predefined time-deadline* T_M*, when the* n*-servers in a DCS perform a synchronous DTR action at the time* $\xi \geq 0$*, satisfies the system of recursive, coupled integral equations in* ξ*:*

$$
\begin{aligned}
Q_\ell(\xi, T_M, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) = \int_0^\xi \Bigg[& \sum_{i=1}^{n} G_{W_{i1}}(\alpha) Q_{\ell_i}\Big(\xi - \alpha, T_M - \alpha, (\mathbf{a}_M + \alpha)^{(i)}, \\
& \mathbf{a}_F + \alpha, \mathbf{a}_C + \alpha\Big) + \sum_{i=1}^{n} Q_{\ell_i'}\Big(\xi - \alpha, T_M - \alpha, \mathbf{a}_M + \alpha, (\mathbf{a}_F + \alpha)^{(ii)}, \mathbf{a}_{C_0} + \alpha\Big) \\
& \times G_{Y_i}(\alpha) + \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} G_{X_{ij}}(\alpha) Q_{\ell_{ij}}\Big(\xi - \alpha, T_M - \alpha, \mathbf{a}_M + \alpha, \mathbf{a}_F + \alpha, \mathbf{a}_C + \alpha\Big) \Bigg] d\alpha \\
& + \big(1 - F_{\tau_{\mathbf{a}}}(\xi)\big) Q_\ell(0, T_M - \xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C), \qquad (3.7)
\end{aligned}
$$

where recursions are carried out in the discrete variables $\mathbf{m}$ *and* $\mathbf{F}$*, the vectors* $\mathbf{m}$ *and* $\mathbf{a}_M$ *and the matrices* $\mathbf{F}$*,* $\mathbf{0}$*,* $\mathbf{a}_F$*, and* $\mathbf{a}_C$ *denote an arbitrarily specified initial system configuration,* $\ell = h(\mathbf{m}, \mathbf{F}, \mathbf{0})$*,* $\ell_i = h(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{F}, \mathbf{0})$*,* $\ell_i' = h(\mathbf{m}, \mathbf{F}^{(ii)}, \mathbf{0})$*,* $\ell_{ij} = h(\mathbf{m}, \mathbf{F}^{(ji)}, \mathbf{0})$*, the vector* $\mathbf{v}^{(i)}$ *(respectively, the matrix* $\mathbf{A}^{(ij)}$*) is identical to the vector* $\mathbf{v}$ *(respectively, the matrix* $\mathbf{A}$*) but with its* i*th (respectively,* ij*th) component set to zero, and* $Q_\ell(0, T_M - \xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ *is the initial condition related to the* ℓ*th integral equation.*

Proof: See Appendix A.

Corollary 1 (Age-dependent characterization of the service reliability)**.** *Consider an n-server DCS whose servers perform a synchronous DTR action at the time $\xi \geq 0$. For any $\ell \in \mathcal{I}$, the service reliability in executing an application, when the n-servers in a DCS perform a synchronous DTR action at the time $\xi \geq 0$, satisfies the system of recursive, coupled integral equations in ξ:*

$$\begin{aligned} R_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) = \int_0^\xi \Bigg[& \sum_{i=1}^{n} G_{W_{i1}}(\alpha) R_{\ell_i}\Big(\xi - \alpha, (\mathbf{a}_M + \alpha)^{(i)}, \mathbf{a}_F + \alpha, \mathbf{a}_C + \alpha\Big) \\ & + \sum_{i=1}^{n} G_{Y_i}(\alpha) R_{\ell'_i}\Big(\xi - \alpha, \mathbf{a}_M + \alpha, (\mathbf{a}_F + \alpha)^{(ii)}, \mathbf{a}_{C_0} + \alpha\Big) \\ & + \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} G_{X_{ij}}(\alpha) R_{\ell_{ij}}\Big(\xi - \alpha, \mathbf{a}_M + \alpha, \mathbf{a}_F + \alpha, \mathbf{a}_C + \alpha\Big) \Bigg] d\alpha \\ & + \big(1 - F_{\tau_{\mathbf{a}}}(\xi)\big) R_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C), \end{aligned} \tag{3.8}$$

where recursions are carried out in the discrete variables $\mathbf{m}$ and $\mathbf{F}$, the vectors $\mathbf{m}$ and $\mathbf{a}_M$ and the matrices $\mathbf{F}$, $\mathbf{0}$, $\mathbf{a}_F$, and $\mathbf{a}_C$ denote an arbitrarily specified initial system configuration, $\ell = h(\mathbf{m}, \mathbf{F}, \mathbf{0})$, $\ell_i = h(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{F}, \mathbf{0})$, $\ell'_i = h(\mathbf{m}, \mathbf{F}^{(ii)}, \mathbf{0})$, $\ell_{ij} = h(\mathbf{m}, \mathbf{F}^{(ji)}, \mathbf{0})$, the vector $\mathbf{v}^{(i)}$ (respectively, the matrix $\mathbf{A}^{(ij)}$) is identical to the vector $\mathbf{v}$ (respectively, the matrix $\mathbf{A}$) but with its ith (respectively, ijth) component set to zero, and $R_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ is the initial condition related to the ℓth integral equation.

In our evaluations, we are interested in predict the metrics the average execution time, QoS and service reliability, for a given DCS with an arbitrarily specified initial system configuration $\mathbf{S}_0 = \mathbf{S}(0) = (\mathbf{m}_0, \mathbf{F}_0, \mathbf{C}_0, \mathbf{a}_{M_0}, \mathbf{a}_{F_0}, \mathbf{a}_{C_0})$. In this dissertation we have assumed that at time $t = 0$, the system configuration is the following: $\mathbf{S}_0$ is $\mathbf{m}_0 = (m_1, \ldots, m_n)^T$ meaning that there is an initial allocation of tasks onto the servers, $\mathbf{F}_0 = \mathbf{1}$ (an all-ones matrix) meaning that all the servers are assumed to be functioning, $\mathbf{C}_0 = \mathbf{0}$ (the null-matrix) meaning that the DTR policy has not been executed yet by the servers, and $\mathbf{a}_0 = \mathbf{0}$ because it is supposed that, at $t = 0$, all the random times have age zero. Thus, in our evaluations we will calculate the metrics:

$\overline{T}_{\ell_0}(\xi, \mathbf{0}, \mathbf{0}, \mathbf{0})$, $Q_{\ell_0}(\xi, T_M, \mathbf{0}, \mathbf{0}, \mathbf{0})$, and $R_{\ell_0}(0, \mathbf{0}, \mathbf{0}, \mathbf{0})$, with $\ell_0 = h(\mathbf{m}_0, \mathbf{F}_0, \mathbf{C}_0)$.

To solve the recursions of Theorems 5 and 6 and Corollary 1, the following values must be specified: (i) the configuration of the system at time $t = 0$; (ii) some particular configurations for which the values of $\overline{T}_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$, $Q_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$, and $R_\ell(\xi, \mathbf{a}_F, \mathbf{a}_C)$ are known; (iii) the configuration of the system at time $t = t_b$ when the servers execute a predetermined DTR policy $\mathbf{L}$; and (iv) the values for the initial conditions $\overline{T}_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$, $Q_\ell(0, T_M - \xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$, and $R_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$.

It was already stated that at time $t = 0$, the system configuration is $\mathbf{S}_0 = (\mathbf{m}_0, \mathbf{F}_0, \mathbf{C}_0, \mathbf{a}_{M_0}, \mathbf{a}_{F_0}, \mathbf{a}_{C_0})$, where $\mathbf{m}_0 = (m_1, \ldots, m_n)^T$, $\mathbf{F}_0 = \mathbf{1}$ (an all-ones matrix) since all the servers are assumed to be functioning, $\mathbf{C}_0 = \mathbf{0}$ (the null-matrix) since the DTR policy has not been executed yet by the servers, and $\mathbf{a}_0 = \mathbf{0}$ because it is supposed that, at $t = 0$, all the random times have age zero.

At any time, the values taken by the metrics are known in the following particular cases for any ξ and ℓ: (i) if the application is composed of a single task queued at the ith server then $\overline{T}_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) = \mathsf{E}[W_{i1}]$; (ii) if there are no tasks to be served in the DCS then $\overline{T}_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ is equal to zero, while $Q_\ell(\xi, T_M - \alpha, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ and $R_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ are both equal to one; (iii) if a server fails and contains at least one task in its queue then $\overline{T}_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ is infinite, while $Q_\ell(\xi, T_M - \alpha, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ and $R_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ are both equal to zero.

Finally, the values of the initial conditions $\overline{T}_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$, $Q_\ell(0, T_M - \xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ and $R_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ can be computed using the results of Theorems 7, 8 and Corollary 2, respectively, and recalling that due the execution of the DTR policy at time $t = t_b$, the new system configuration is $\mathbf{S}_{t_b} = \mathbf{S}(t_b) = (\mathbf{m}_{t_b}, \mathbf{F}_{t_b}, \mathbf{C}_{t_b}, \mathbf{a}_{M_{t_b}}, \mathbf{a}_{F_{t_b}}, \mathbf{a}_{C_{t_b}})$, where $\mathbf{m}_{t_b} = (r_1', \ldots, r_n')^T$ where $r_i' = r_i - \sum_{j=1, j\neq i}^{n} l_{ij}$, $\mathbf{F}_{t_b} = \mathbf{F}$, $\mathbf{C}_{t_b} = \mathbf{L}$, $\mathbf{a}_{M_{t_b}} = \mathbf{a}_M + t_b$, $\mathbf{a}_{F_{t_b}} = \mathbf{a}_F + t_b$, the ages of the elements of $\mathbf{a}_C$ associated with the tasks being transferred in the network are set to zero.

Theorem 7 (Initial condition for the age-dependent characterization of the average execution time)**.** *Consider an n-server DCS with an arbitrarily specified initial system configuration* $\mathbf{m}$*,* $\mathbf{F}$*,* $\mathbf{C}$*,* $\mathbf{a}_M$*,* $\mathbf{a}_F$*, and* $\mathbf{a}_C$*. For any* $\ell \in \mathcal{I}$*, the initial condition* $\overline{T}_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ *related to the* ℓ*th integral equation of the characterization for the average application execution time satisfies the system of recursive, coupled integral equations:*

$$\overline{T}_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) = \mathsf{E}[\tau_{\mathbf{a}}] + \int_0^\infty \Bigg[\sum_{i=1}^{n} G_{W_{i1}}(\alpha) \overline{T}_{\ell_i}\big(0, (\mathbf{a}_M+\alpha)^{(i)}, \mathbf{a}_F+\alpha, \mathbf{a}_C+\alpha\big) + \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} G_{Z_{ji}}(\alpha) \overline{T}_{\ell'_{ji}}\Big(0, \mathbf{a}_M+\alpha, \mathbf{a}_F+\alpha, \mathbf{a}_C+\alpha\Big) \Bigg] d\alpha, \quad (3.9)$$

where recursions are carried out in the discrete variables $\mathbf{m}$*,* $\mathbf{F}$*, and* $\mathbf{C}$*, the vectors* $\mathbf{m}$ *and* $\mathbf{a}_M$ *and the matrices* $\mathbf{F}$*,* $\mathbf{C}$*,* $\mathbf{a}_F$*, and* $\mathbf{a}_C$ *denote an arbitrarily specified initial system configuration,* $\ell = h(\mathbf{m}, \mathbf{F}, \mathbf{C})$*,* $\ell_i = h(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{F}, \mathbf{C})$*,* $\ell'_{ji} = h(\mathbf{m}+c_{ji}\boldsymbol{\delta}_i, \mathbf{F}, \mathbf{C}^{(ji)})$*, and the vector* $\mathbf{v}^{(i)}$ *(respectively, the matrix* $\mathbf{A}^{(ij)}$*) is identical to the vector* $\mathbf{v}$ *(respectively, the matrix* $\mathbf{A}$*) but with its* i*th (respectively,* ij*th) component set to zero.*

The proof of Theorem 7 is sketched in Appendix B since it is similar to that of Theorems 5 and 6. The reader is referred to [70] for a proof in the special case of $n = 2$ servers.

Theorem 8 (Initial condition for the age-dependent characterization of the QoS)**.** *Consider an n-server DCS with an arbitrarily specified initial system configuration* $\mathbf{m}$*,* $\mathbf{F}$*,* $\mathbf{C}$*,* $\mathbf{a}_M$*,* $\mathbf{a}_F$*, and* $\mathbf{a}_C$*. For any* $\ell \in \mathcal{I}$*, the initial condition* $Q_\ell(0, T'_M, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$ *related to the* ℓ*th integral equation of the characterization for the QoS in executing*

an application satisfies the system of coupled integral recursions:

$$
\begin{aligned}
Q_\ell(0,T'_M,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C) = \int_0^\infty \Bigg[& \sum_{i=1}^{n} G_{W_{i1}}(\alpha) Q_{\ell_i}\Big(0, T'_M-\alpha, (\mathbf{a}_M+\alpha)^{(i)}, \mathbf{a}_F+\alpha, \\
& \mathbf{a}_C+\alpha\Big) + \sum_{i=1}^{n} G_{Y_i}(\alpha) Q_{\ell'_i}\Big(0, T'_M-\alpha, \mathbf{a}_M+\alpha, (\mathbf{a}_F+\alpha)^{(ii)}, \mathbf{a}_{C_0}+\alpha\Big) \\
& + \sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n} G_{Z_{ji}}(\alpha)\, Q_{\ell_{ji}}\Big(0, T'_M-\alpha, \mathbf{a}_M+\alpha, \mathbf{a}_F+\alpha, \mathbf{a}_C+\alpha\Big) \\
& + \sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n} G_{X_{ij}}(\alpha) Q_{\ell_{ij}}\Big(0, T'_M-\alpha, \mathbf{a}_M+\alpha, \mathbf{a}_F+\alpha, \mathbf{a}_C+\alpha\Big)\Bigg] d\alpha, \qquad (3.10)
\end{aligned}
$$

where recursions are carried out in the discrete variables $\mathbf{m}$*,* $\mathbf{F}$*, and* $\mathbf{C}$*, the vectors* $\mathbf{m}$ *and* $\mathbf{a}_M$ *and the matrices* $\mathbf{F}$*,* $\mathbf{C}$*,* $\mathbf{a}_F$*, and* $\mathbf{a}_C$ *denote an arbitrarily specified initial system configuration,* $T'_M = T_M - \xi$*,* $\ell = h(\mathbf{m},\mathbf{F},\mathbf{C})$*,* $\ell_i = h(\mathbf{m}-\boldsymbol{\delta}_i,\mathbf{F},\mathbf{C})$*,* $\ell'_i = h(\mathbf{m},\mathbf{F}^{(ii)},\mathbf{C})$*,* $\ell_{ij} = h(\mathbf{m},\mathbf{F}^{(ji)},\mathbf{C})$*,* $\ell'_{ji} = h(\mathbf{m}+c_{ji}\boldsymbol{\delta}_i,\mathbf{F},\mathbf{C}^{(ji)})$*, and the vector* $\mathbf{v}^{(i)}$ *(respectively, the matrix* $\mathbf{A}^{(ij)}$*) is identical to the vector* $\mathbf{v}$ *(respectively, the matrix* $\mathbf{A}$*) but with its* i*th (respectively,* ij*th) component set to zero.*

The proof of Theorem 8 is sketched in Appendix B since it is similar to that of Theorems 5 and 6. The reader is referred to [70] for a proof in the special case of $n = 2$ servers.

Corollary 2 (Initial condition for the age-dependent characterization of the service reliability)**.** *Consider an* n*-server DCS with an arbitrarily specified initial system configuration* $\mathbf{m}$*,* $\mathbf{F}$*,* $\mathbf{C}$*,* $\mathbf{a}_M$*,* $\mathbf{a}_F$*, and* $\mathbf{a}_C$*. For any* $\ell \in \mathcal{I}$*, the initial condition* $R_\ell(0,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C)$ *related to the* ℓ*th integral equation of the characterization for the*

QoS in executing an application satisfies the system of coupled integral recursions:

$$
\begin{aligned}
R_\ell(0,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C) = \int_0^\infty \Bigg[& \sum_{i=1}^{n} G_{W_{i1}}(\alpha) R_{\ell_i}\Big(0,(\mathbf{a}_M+\alpha)^{(i)},\mathbf{a}_F+\alpha,\mathbf{a}_C+\alpha\Big) \\
& + \sum_{i=1}^{n} G_{Y_i}(\alpha) R_{\ell_i'}\Big(0,\mathbf{a}_M+\alpha,(\mathbf{a}_F+\alpha)^{(ii)},\mathbf{a}_{C_0}+\alpha\Big) \\
& + \sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n} G_{Z_{ji}}(\alpha)\, R_{\ell_{ji}}\Big(0,\mathbf{a}_M+\alpha,\mathbf{a}_F+\alpha,\mathbf{a}_C+\alpha\Big) \\
& + \sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n} G_{X_{ij}}(\alpha) R_{\ell_{ij}}\Big(0,\mathbf{a}_M+\alpha,\mathbf{a}_F+\alpha,\mathbf{a}_C+\alpha\Big)\Bigg]\, d\alpha,
\end{aligned}
\tag{3.11}
$$

where recursions are carried out in the discrete variables $\mathbf{m}$*,* $\mathbf{F}$*, and* $\mathbf{C}$*, the vectors* $\mathbf{m}$ *and* $\mathbf{a}_M$ *and the matrices* $\mathbf{F}$*,* $\mathbf{0}$*,* $\mathbf{a}_F$*, and* $\mathbf{a}_C$ *denote an arbitrarily specified initial system configuration,* $\ell = h(\mathbf{m},\mathbf{F},\mathbf{C})$*,* $\ell_i = h(\mathbf{m}-\boldsymbol{\delta}_i,\mathbf{F},\mathbf{C})$*,* $\ell_i' = h(\mathbf{m},\mathbf{F}^{(ii)},\mathbf{C})$*,* $\ell_{ij} = h(\mathbf{m},\mathbf{F}^{(ji)},\mathbf{C})$*,* $\ell_{ji}' = h(\mathbf{m}+c_{ji}\boldsymbol{\delta}_i,\mathbf{F},\mathbf{C}^{(ji)})$*, and the vector* $\mathbf{v}^{(i)}$ *(respectively, the matrix* $\mathbf{A}^{(ij)}$*) is identical to the vector* $\mathbf{v}$ *(respectively, the matrix* $\mathbf{A}$*) but with its* i*th (respectively,* ij*th) component set to zero.*

Algorithm to solve the integral recursions

To calculate the performance and reliability metrics the initial task allocation, the t_b instant and a specific DTR policy must be defined. (See Section 4.1.1 for the class of DTR policies used in this work.) In addition, to compute the QoS and the service reliability the value of T_M must be provided. All these conditions, as well as equations (3.6) to (3.10) and the known values for QoS and average execution time in the special configurations discussed in the previous section have been employed in the development of Algorithm 1 that solves the integral recursions.

The algorithm is divided in two stages. The first stage (first while loop in Algorithm 1) dynamically constructs the mapping $h : \Omega \to \mathcal{I}$ by saving a list of all the possible states that the DCS visits as part of its dynamics given that its initial con-

Algorithm 1 Algorithm to solve the integral recursions in Theorems 5 to 8.

Require: $\mathbf{S}_0$, t_b, $\mathbf{L}$, and T_M

Ensure: $\overline{T}(\mathbf{S}_0)$ and $R_{T_M}(\mathbf{S}_0)$

```
KnownValues=⎧T̄(S) = E[W_i1 − a_Mi], R_TM(S′) = 1, R_TM(S′) = 0⎫
ListOfStatesK=∅, ListOfStatesKMinus1=S_0, k=0
while true do {Generate list of possible states}
  CtrStates=0, CtrKnownValues=0
  for OldState ∈ ListOfStatesKMinus1 do
    for X ∈ {W_i1, Y_i, X_ij, Z_ij} do
      NewState ← Change(OldState,X)
      ListOfStatesK ← ListOfStatesK ∪ NewState
      CtrStates=CtrStates+1
      if NewState ∈ KnownValues then
        CtrKnownValues=CtrKnownValues+1
      end if
    end for
  end for
  Store(ListOfStatesK)
  if CtrStates==CtrKnownValues then
    false
  end if
  k←k+1, ListOfStatesK=∅, ListOfStatesKMinus1=ListOfStatesK
end while
while k≠ 0 do {Solving recursions}
  Load(ListOfStatesK)
  Solve (3.6) to (3.10) using ListOfStatesKMinus1 to obtain ListOfStatesK
  ListOfStatesKMinus1=ListOfStatesK, k← k-1
end while
```

figuration at $t = 0$ is as specified by $\mathbf{S}_0$. Example of the recursions generated from a generic initial state $\mathbf{S}_0$. Figure 3.1 shows a simplified case for the the recursions generated from a generic initial state $\mathbf{S}_0$. Comparing Figs. 2.1 and 3.1 it becomes evident that the Markovian model generates a reduced number of states as compared to the non-Markovian case. The second stage, (second while loop in Algorithm 1) solves the recursions in (3.6) to (3.10) by means of the values for the metrics calculated in the previous recursion. This algorithm has been coded in C with the help of the GNU's Scientific Library (GSL), [36], which provides exceptional tools for numerical integration as well as vector and matrix data-type handling. The major challenge in coding Algorithm 1 is the large amount of memory and computational resources it takes due to the construction, sorting and storage of all the state values. Because of memory limitations, the load and store operations of the algorithm must be carried out using the hard drive.

3.2.3 Approximations and bounds

Executing Algorithm 1 to solve the system of recurrence equations is computationally expensive for large DCSs because of two reasons. First, since the dimension of the age-dependent state vector $\mathbf{S}$ is $4n^2 + 2n$, the age-dependent state-space model for the execution time scales polynomially in the number of servers. Second, for a fixed number of servers in a DCS, since both FN packets and tasks exchanged among the servers may arrive at any random instant, every recursion generated from Theorems 5 to 8 and Corollaries 1 and 2 must consider a combinatorial number of values for the age-dependent state-vector, and as a consequence, the number of recursive integral equations that must be constructed to calculate the metrics in Theorems 5 to 8 and Corollaries 1 and 2 scales exponentially in the number of messages exchanged.

In order to circumvent the scalability problem of the age-dependent model, two

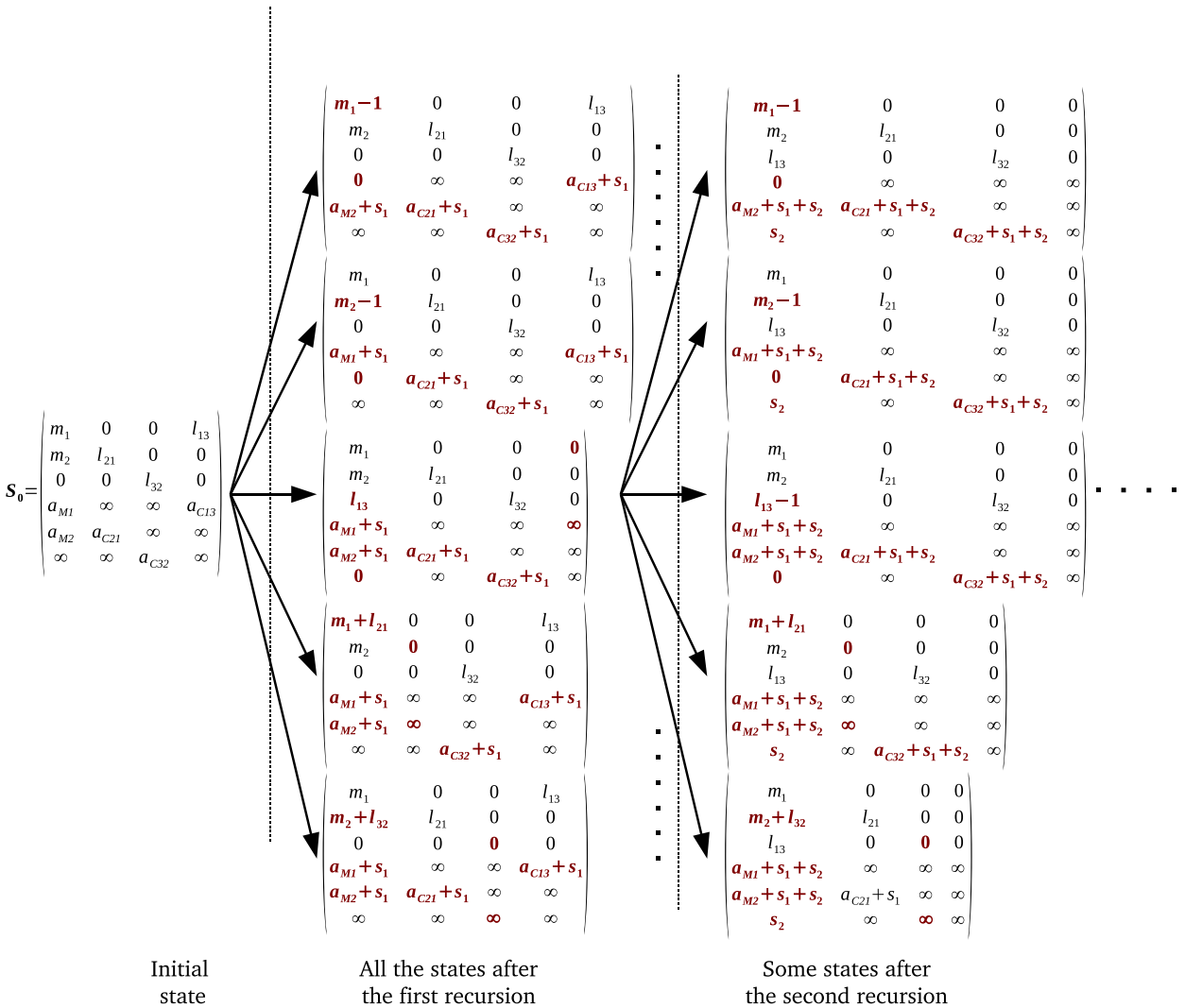


Figure 3.1: Example of the recursions generated from a generic initial state $\mathbf{S}_0$ in the case of a non-Markovian DCS.

simplifying assumptions have been considered in constructing an approximated characterization for the dynamics of the DCS. This approximation has been used to yield approximations for the metrics characterized in Theorems 5 to 8 and Corollaries 1 and 2. The first assumption considered is to neglect the random transfer time of FN packets on the network. This approximation reduces the matrix $\mathbf{F}$ to a binary vector that we have denoted as $\mathbf{f}$. The second assumption considered is to suppose that all

the tasks in transit to a particular server arrive simultaneously to it. More precisely, it has been assumed here that the $l_k = \sum_j l_{jk}$ tasks being transferred from other servers to the kth server arrive simultaneously to server k at a certain random time denoted as $\widetilde{Z}_k$. This approximation reduces the matrix $\mathbf{C}$ to a vector that we have denoted as $\mathbf{c}$. Once these approximations are imposed, the same principles presented in Section 3.2.1 can be exploited to obtain a set of regenerative age-dependent equations for the approximated random execution time. To this end, we must consider the following extra assumption.

Assumption A3. Suppose that the pdf of the random times $\widetilde{Z}_i$ are known and denoted as $f_{\widetilde{Z}_i}(t)$. Suppose also that these random times are mutually independent and also are mutually independent to all the random times listed in Assumptions A1.

In addition, we associate the age-variables a_{C_i} to the random times $\widetilde{Z}_i$, and, as in the case of the complete characterizations for performance and reliability, we note that the vectors $\mathbf{m}$, $\mathbf{f}$ and $\mathbf{c}$ have finite dimensions and take values on the finite discrete sets $\Omega_1 = \{0, 1, \ldots, M\}^n$, $\widetilde{\Omega}_2 = \{0, 1\}^n$ and $\widetilde{\Omega}_3 = \{0, M\}^n$, respectively. Thus, for $\widetilde{\Omega} = \Omega_1 \times \widetilde{\Omega}_2 \times \widetilde{\Omega}_3$, we can always define any one-to-one mapping $\widetilde{h} : \widetilde{\Omega} \to \widetilde{\mathcal{I}}$ such that, for each possible value of the concatenated vector $(\mathbf{m}, \mathbf{f}, \mathbf{c})$ in $\widetilde{\Omega}$, the mapping $\widetilde{h}(\mathbf{m}, \mathbf{f}, \mathbf{c})$ assigns a positive integer in the index set $\widetilde{\mathcal{I}} = \{1, 2, \ldots, \widetilde{\kappa}\}$, where $\widetilde{\kappa}$ is the cardinality of Ω.

Empowered by these approximations, we introduce the *approximated age-dependent random execution time of an application*, $T_\ell(t_b, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C)$, when the system configuration is as specified by the reduced age-dependent state vector $\widetilde{\mathbf{S}} = (\mathbf{m}, \mathbf{f}, \mathbf{c}, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C)$, where $\ell = \widetilde{h}(\mathbf{m}, \mathbf{f}, \mathbf{c})$. It must be noted that these approximations reduce the dimension of the age-dependent state vector to $6n$, yielding a model with linear scalability in the number of servers.

The rationale for approximating the random task-arrival times can be further exploited to define upper and lower bounds for the random execution time of an application. To do so, note that the random execution time of an application is the maximum over all the random execution times of each server. In addition, the random execution time of each server is equal to the random service time of the tasks allocated at the server's queue plus the random time taken to receive and serve the tasks in transit to itself. Therefore, it can be concluded that a lower bound (consequently, an upper bound) for the random execution time of an application can be obtained by assuming that all the tasks arrive simultaneously to all the recipient servers at the lowest (consequently, at the highest) random task-transfer time occurring in the DCS. More precisely, the overall minimum and maximum random task-transfer times can be defined as $Z^{\min} = \min_{i,j} Z_{ij}$ and $Z^{\max} = \max_{i,j} Z_{ij}$, respectively. Note that this new approximation reduces the vector $\mathbf{c}$ to a constant, c. In order to exploit, one more time, the principles of stochastic regeneration to obtain a set of regenerative age-dependent equations for the lower and upper bounds of the execution time, the following extra assumption is required.

Assumption A4. Suppose that the pdf of the random times $Z^{\min}$ and $Z^{\max}$ are known and denoted, respectively, as $f_{Z^{\min}}(t)$ and $f_{Z^{\max}}(t)$. Suppose also that these random times are mutually independent and also are mutually independent to all the random times listed in Assumptions A1.

Additionally, we must associate the age-variables $a_{C_{\min}}$ and $a_{C_{\max}}$ to the random times $Z^{\min}$ and $Z^{\max}$, respectively. Noting one more time that the vectors $\mathbf{m}$ and $\mathbf{f}$ and the constant c have finite dimensions and take values on the finite discrete sets Ω_1, $\widetilde{\Omega}_2$ and $\widetilde{\Omega}'_3 = \{0, M\}$, respectively. Thus, for $\widetilde{\Omega}' = \Omega_1 \times \widetilde{\Omega}_2 \times \widetilde{\Omega}'_3$, we can always define any one-to-one mapping $\widetilde{g} : \widetilde{\Omega}' \to \widetilde{\mathcal{I}}'$ such that, for each possible value of the concatenated vector $(\mathbf{m}, \mathbf{f}, c)$ in $\widetilde{\Omega}'$, the mapping $\widetilde{g}(\mathbf{m}, \mathbf{f}, c)$ assigns a positive integer in the index set $\widetilde{\mathcal{I}} = \{1, 2, \ldots, \widetilde{\kappa}'\}$, where $\widetilde{\kappa}'$ is the cardinality of $\widetilde{\Omega}'$.

Now we can introduce the *lower and the upper bounds for the random execution time of an application*, respectively, as $T_\ell^{\min}(t_b, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{a}_C)$ and $T_\ell^{\max}(t_b, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{a}_C)$, when the system configuration is as specified by the reduced age-dependent state vector $\widetilde{\mathbf{S}}' = (\mathbf{m}, \mathbf{f}, c, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{a}_C)$, where $\ell = \widetilde{g}(\mathbf{m}, \mathbf{f}, \mathbf{c})$.

For the sake of the space, characterizations only for the approximation and the lower bound of the performance and reliability metrics are presented. The structure of the remaining set of equations is similar to the ones presented here and in Theorems 5 to 8 and Corollaries 1 and 2.

Theorem 9 (Age-dependent characterization for the approximated average execution time)**.** *Consider an n-server DCS whose servers perform a synchronous DTR action at the time $\xi \geq 0$. For any $\ell \in \mathcal{I}$, the average approximated execution time of an application satisfies the system of recursive, coupled integral equations in ξ:*

$$\overline{T}_\ell(\xi, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C) = \int_0^\xi \sum_{i=1}^{n} G_{W_{i1}}(\alpha) \overline{T}_{\ell_i}\big(\xi - \alpha, (\widetilde{\mathbf{a}}_M + \alpha)^{(i)}, \widetilde{\mathbf{a}}_F + \alpha, \widetilde{\mathbf{a}}_C + \alpha\big)\, d\alpha + \int_0^\xi \alpha f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha + \big(1 - F_{\tau_{\mathbf{a}}}(\xi)\big)\big(\xi + \overline{T}_\ell(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C)\big), \tag{3.12}$$

where recursions are carried out in the discrete variables $\mathbf{m}$ and $\mathbf{f}$, the vectors $\mathbf{m}$, $\mathbf{f}$, $\mathbf{0}$, $\widetilde{\mathbf{a}}_M$, $\widetilde{\mathbf{a}}_F$, and $\widetilde{\mathbf{a}}_C$ denote an arbitrarily specified initial system configuration, $\ell = \widetilde{h}(\mathbf{m}, \mathbf{f}, \mathbf{0})$, $\ell_i = \widetilde{h}(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{f}, \mathbf{0})$, $\boldsymbol{\delta}_i$ denotes an n-dimensional vector with all its entries equal to zero except that its ith element is equal to one, the vector $\mathbf{v}^{(i)}$ is identical to the vector $\mathbf{v}$ but with its ith component set to zero, and $\overline{T}_\ell(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C)$ is the initial condition related to the ℓth integral equation. Moreover, these initial conditions satisfy the system of recursive, coupled integral equations:

$$\overline{T}_{\ell'}(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C) = \mathsf{E}[\tau_{\mathbf{a}}] + \int_0^\infty \left[\sum_{i=1}^{n} G_{W_{i1}}(\alpha) \overline{T}_{\ell_i}\big(0, (\widetilde{\mathbf{a}}_M + \alpha)^{(i)}, \widetilde{\mathbf{a}}_F + \alpha, \widetilde{\mathbf{a}}_C + \alpha\big) + \sum_{i=1}^{n} G_{\widetilde{Z}_i}(\alpha) \overline{T}_{\ell'_i}\Big(0, \widetilde{\mathbf{a}}_M + \alpha, \widetilde{\mathbf{a}}_F + \alpha, \widetilde{\mathbf{a}}_C + \alpha\Big) \right] d\alpha, \tag{3.13}$$

where recursions are carried out in the discrete variables $\mathbf{m}$ *and* $\mathbf{f}$*, the vectors* $\mathbf{m}$*,* $\mathbf{f}$*,* $\mathbf{c}$*,* $\widetilde{\mathbf{a}}_M$*,* $\widetilde{\mathbf{a}}_F$*, and* $\widetilde{\mathbf{a}}_C$ *denote an arbitrarily specified initial system configuration,* $\ell' = \widetilde{h}(\mathbf{m}, \mathbf{f}, \mathbf{c})$*, and* $\ell'_i = \widetilde{h}(\mathbf{m} + l_i\boldsymbol{\delta}_i, \mathbf{f}, \mathbf{c}^i)$*.*

Theorem 10 (Age-dependent characterization for the approximated QoS)**.** *Consider an n-server DCS whose servers perform a synchronous DTR action at the time* $\xi \geq 0$*. For any* $\ell \in \mathcal{I}$*, the approximated QoS in executing an application satisfies the system of recursive, coupled integral equations in* ξ*:*

$$Q_\ell(\xi,T_M,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C) = \int_0^\xi \Bigg[\sum_{i=1}^{n} G_{W_{i1}}(\alpha) Q_{\ell_i}\Big(\xi-\alpha, T_M-\alpha, (\widetilde{\mathbf{a}}_M+\alpha)^{(i)}, \widetilde{\mathbf{a}}_F+\alpha, \widetilde{\mathbf{a}}_C+\alpha\Big) + \sum_{i=1}^{n} G_{Y_i}(\alpha) Q_{\ell'_i}\Big(\xi-\alpha, T_M-\alpha, \widetilde{\mathbf{a}}_M+\alpha, (\widetilde{\mathbf{a}}_F+\alpha)^{(ii)}, \widetilde{\mathbf{a}}_C+\alpha\Big)\Bigg] d\alpha + \big(1-F_{\tau_{\mathbf{a}}}(\xi)\big) Q_\ell(0, T_M-\xi, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C), \tag{3.14}$$

where recursions are carried out in the discrete variables $\mathbf{m}$ *and* $\mathbf{f}$*, the vectors* $\mathbf{m}$*,* $\mathbf{f}$*,* $\mathbf{0}$*,* $\widetilde{\mathbf{a}}_M$*,* $\widetilde{\mathbf{a}}_F$*, and* $\widetilde{\mathbf{a}}_C$ *denote an arbitrarily specified initial system configuration,* $\ell = \widetilde{h}(\mathbf{m}, \mathbf{f}, \mathbf{0})$*,* $\ell_i = \widetilde{h}(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{f}, \mathbf{0})$*,* $\ell'_i = \widetilde{h}(\mathbf{m}, \mathbf{f}^i, \mathbf{0})$*,* $\boldsymbol{\delta}_i$ *denotes an n-dimensional vector with all its entries equal to zero except that its ith element is equal to one, the vector* $\mathbf{v}^{(i)}$ *is identical to the vector* $\mathbf{v}$ *but with its ith component set to zero, and* $Q_\ell(0, T_M-\xi, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C)$ *is the initial condition related to the* ℓ*th integral equation. Moreover, these initial conditions satisfy the system of recursive, coupled integral equations:*

$$Q_{\ell'}(0,T'_M,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C) = \int_0^\infty \Bigg[\sum_{i=1}^{n} G_{W_{i1}}(\alpha) Q_{\ell_i}\Big(0, T'_M-\alpha, (\widetilde{\mathbf{a}}_M+\alpha)^{(i)}, \widetilde{\mathbf{a}}_F+\alpha, \widetilde{\mathbf{a}}_C+\alpha\Big) + \sum_{i=1}^{n} G_{Y_i}(\alpha) Q_{\ell'_i}\Big(0, T'_M-\alpha, \widetilde{\mathbf{a}}_M+\alpha, (\widetilde{\mathbf{a}}_F+\alpha)^{(ii)}, \widetilde{\mathbf{a}}_{C_0}+\alpha\Big) + \sum_{i=1}^{n} G_{\widetilde{Z}_i}(\alpha)\, Q_{\ell''_i}\Big(0, T'_M-\alpha, \widetilde{\mathbf{a}}_M+\alpha, \widetilde{\mathbf{a}}_F+\alpha, \widetilde{\mathbf{a}}_C+\alpha\Big)\Bigg] d\alpha, \tag{3.15}$$

where recursions are carried out in the discrete variables $\mathbf{m}$*,* $\mathbf{f}$*, and* $\mathbf{c}$*, the vectors* $\mathbf{m}$*,* $\mathbf{f}$*,* $\mathbf{c}$*,* $\widetilde{\mathbf{a}}_M$*,* $\widetilde{\mathbf{a}}_F$*, and* $\widetilde{\mathbf{a}}_C$ *denote an arbitrarily specified initial system configuration,* $T'_M =$

$T_M - \xi$, $\ell' = \widetilde{h}(\mathbf{m}, \mathbf{f}, \mathbf{c})$, $\ell_i = \widetilde{h}(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{F}, \mathbf{C})$, $\ell_i' = \widetilde{h}(\mathbf{m}, \mathbf{f}^{(i)}, \mathbf{c})$, $\ell_i'' = h(\mathbf{m} + l_i\boldsymbol{\delta}_i, \mathbf{c}, \mathbf{c}^{(i)})$, *and the vector* $\mathbf{v}^{(i)}$ *is identical to the vector* $\mathbf{v}$ *but with its* i*th component set to zero.*

Corollary 3 (Age-dependent characterization for the approximated service reliability)**.** *Consider an* n*-server DCS whose servers perform a synchronous DTR action at the time* $\xi \geq 0$*. For any* $\ell \in \mathcal{I}$*, the approximated service reliability in executing an application satisfies the system of recursive, coupled integral equations in* ξ*:*

$$
\begin{aligned}
R_\ell(\xi, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C) = \int_0^\xi \Bigg[& \sum_{i=1}^{n} G_{W_{i1}}(\alpha) R_{\ell_i}\Big(\xi - \alpha, (\widetilde{\mathbf{a}}_M + \alpha)^{(i)}, \widetilde{\mathbf{a}}_F + \alpha, \widetilde{\mathbf{a}}_C + \alpha\Big) \\
& + \sum_{i=1}^{n} G_{Y_i}(\alpha) R_{\ell_i'}\Big(\xi - \alpha, \widetilde{\mathbf{a}}_M + \alpha, (\widetilde{\mathbf{a}}_F + \alpha)^{(ii)}, \widetilde{\mathbf{a}}_C + \alpha\Big) \Bigg] d\alpha \\
& + \big(1 - F_{\tau_{\mathbf{a}}}(\xi)\big) R_\ell(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C),
\end{aligned}
\tag{3.16}
$$

where recursions are carried out in the discrete variables $\mathbf{m}$ *and* $\mathbf{f}$*, the vectors* $\mathbf{m}$*,* $\mathbf{f}$*,* $\mathbf{0}$*,* $\widetilde{\mathbf{a}}_M$*,* $\widetilde{\mathbf{a}}_F$*, and* $\widetilde{\mathbf{a}}_C$ *denote an arbitrarily specified initial system configuration,* $\ell = \widetilde{h}(\mathbf{m}, \mathbf{f}, \mathbf{0})$*,* $\ell_i = \widetilde{h}(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{f}, \mathbf{0})$*,* $\ell_i' = \widetilde{h}(\mathbf{m}, \mathbf{f}^i, \mathbf{0})$*,* $\boldsymbol{\delta}_i$ *denotes an* n*-dimensional vector with all its entries equal to zero except that its* i*th element is equal to one, the vector* $\mathbf{v}^{(i)}$ *is identical to the vector* $\mathbf{v}$ *but with its* i*th component set to zero, and* $R_\ell(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C)$ *is the initial condition related to the* ℓ*th integral equation. Moreover, these initial conditions satisfy the system of recursive, coupled integral equations:*

$$
\begin{aligned}
R_{\ell'}(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C) = \int_0^\infty \Bigg[& \sum_{i=1}^{n} G_{W_{i1}}(\alpha) R_{\ell_i}\Big(0, (\widetilde{\mathbf{a}}_M + \alpha)^{(i)}, \widetilde{\mathbf{a}}_F + \alpha, \widetilde{\mathbf{a}}_C + \alpha\Big) \\
& + \sum_{i=1}^{n} G_{Y_i}(\alpha) R_{\ell_i'}\Big(0, \widetilde{\mathbf{a}}_M + \alpha, (\widetilde{\mathbf{a}}_F + \alpha)^{(ii)}, \widetilde{\mathbf{a}}_{C_0} + \alpha\Big) \\
& + \sum_{i=1}^{n} G_{\widetilde{Z}_i}(\alpha)\, R_{\ell_i''}\Big(0, \widetilde{\mathbf{a}}_M + \alpha, \widetilde{\mathbf{a}}_F + \alpha, \widetilde{\mathbf{a}}_C + \alpha\Big) \Bigg] d\alpha,
\end{aligned}
\tag{3.17}
$$

where recursions are carried out in the discrete variables $\mathbf{m}$*,* $\mathbf{f}$*, and* $\mathbf{c}$*, the vectors* $\mathbf{m}$*,* $\mathbf{f}$*,* $\mathbf{c}$*,* $\widetilde{\mathbf{a}}_M$*,* $\widetilde{\mathbf{a}}_F$*, and* $\widetilde{\mathbf{a}}_C$ *denote an arbitrarily specified initial system configuration,*

$\ell' = \widetilde{h}(\mathbf{m}, \mathbf{f}, \mathbf{c})$, $\ell_i = \widetilde{h}(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{f}, \mathbf{c})$, $\ell_i' = \widetilde{h}(\mathbf{m}, \mathbf{f}^{(i)}, \mathbf{c})$, $\ell_i'' = h(\mathbf{m} + l_i\boldsymbol{\delta}_i, \mathbf{f}, \mathbf{c}^{(i)})$, *and the vector* $\mathbf{v}^{(i)}$ *is identical to the vector* $\mathbf{v}$ *but with its ith component set to zero.*

Theorem 11 (Age-dependent characterization for the lower bound of the average execution time)**.** *Consider an n-server DCS whose servers perform a synchronous DTR action at the time* $\xi \geq 0$*. For any* $\ell \in \mathcal{I}$*, the lower bound for the average execution time of an application satisfies the system of recursive, coupled integral equations in* ξ*:*

$$\begin{aligned}\overline{T}_{\ell}^{\min}(\xi, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{a}_C) = &\int_0^{\xi} \sum_{i=1}^{n} G_{W_{i1}}(\alpha)\overline{T}_{\ell_i}^{\min}\big(\xi - \alpha, (\widetilde{\mathbf{a}}_M + \alpha)^{(i)}, \widetilde{\mathbf{a}}_F + \alpha, \widetilde{a}_C + \alpha\big)\, d\alpha \\ &+ \int_0^{\xi} \alpha f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha + \big(1 - F_{\tau_{\mathbf{a}}}(\xi)\big)\big(\xi + \overline{T}_{\ell}^{\min}(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{a}_C)\big),\end{aligned} \tag{3.18}$$

where recursions are carried out in the discrete variables $\mathbf{m}$ *and* $\mathbf{f}$*, the vectors* $\mathbf{m}$*,* $\mathbf{f}$*,* $\widetilde{\mathbf{a}}_M$*, and* $\widetilde{\mathbf{a}}_F$ *and the scalars 0 and* $\widetilde{a}_C$ *denote an arbitrarily specified initial system configuration,* $\ell = \widetilde{g}(\mathbf{m}, \mathbf{f}, 0)$*,* $\ell_i = \widetilde{g}(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{f}, 0)$*,* $\boldsymbol{\delta}_i$ *denotes an n-dimensional vector with all its entries equal to zero except that its ith element is equal to one, the vector* $\mathbf{v}^{(i)}$ *is identical to the vector* $\mathbf{v}$ *but with its ith component set to zero, and* $\overline{T}_{\ell}^{\min}(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{a}_C)$ *is the initial condition related to the* ℓ*th integral equation. Moreover, these initial conditions satisfy the system of recursive, coupled integral equations:*

$$\begin{aligned}\overline{T}_{\ell'}^{\min}(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{a}_C) = \mathsf{E}[\tau_{\mathbf{a}}] + \int_0^{\infty} \Bigg[&\sum_{i=1}^{n} G_{W_{i1}}(\alpha)\overline{T}_{\ell_i}^{\min}\big(0, (\widetilde{\mathbf{a}}_M + \alpha)^{(i)}, \widetilde{\mathbf{a}}_F + \alpha, \widetilde{a}_C + \alpha\big) \\ &+ G_{\widetilde{Z}^{\min}}(\alpha)\overline{T}_{\ell''}^{\min}\Big(0, \widetilde{\mathbf{a}}_M + \alpha, \widetilde{\mathbf{a}}_F + \alpha, \widetilde{a}_C + \alpha\Big)\Bigg]\, d\alpha,\end{aligned} \tag{3.19}$$

where recursions are carried out in the discrete variables $\mathbf{m}$ *and* $\mathbf{f}$*, the vectors* $\mathbf{m}$*,* $\mathbf{f}$*,* $\widetilde{\mathbf{a}}_M$*, and* $\widetilde{\mathbf{a}}_F$ *and the scalars c and* $\widetilde{\mathbf{a}}_C$ *denote an arbitrarily specified initial system configuration,* $\ell' = \widetilde{g}(\mathbf{m}, \mathbf{f}, c)$*, and* $\ell_i = \widetilde{g}(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{f}, 0)$*, and* $\ell'' = \widetilde{g}(\mathbf{m} + \mathbf{l}, \mathbf{f}, 0)$*, where* $\mathbf{l} = (l_1, \ldots, l_n)$ *is a column vector describing the total number of tasks in transit to each server.*

Theorem 12 (Age-dependent characterization for the lower bound of the QoS)**.** *Consider an n-server DCS whose servers perform a synchronous DTR action at the time $\xi \geq 0$. For any $\ell \in \mathcal{I}$, the lower bound of the QoS in executing an application satisfies the system of recursive, coupled integral equations in ξ:*

$$Q_{\ell}^{\min}(\xi,T_M,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{a}_C) = \int_0^{\xi} \Bigg[\sum_{i=1}^{n} G_{W_{i1}}(\alpha) Q_{\ell_i}^{\min}\Big(\xi-\alpha, T_M-\alpha, (\widetilde{\mathbf{a}}_M+\alpha)^{(i)}, \widetilde{\mathbf{a}}_F+\alpha, \widetilde{a}_C+\alpha\Big) + \sum_{i=1}^{n} G_{Y_i}(\alpha) Q_{\ell_i'}^{\min}\Big(\xi-\alpha, T_M-\alpha, \widetilde{\mathbf{a}}_M+\alpha, (\widetilde{\mathbf{a}}_F+\alpha)^{(ii)}, \widetilde{a}_C+\alpha\Big)\Bigg] d\alpha + \big(1-F_{\tau_{\mathbf{a}}}(\xi)\big) Q_{\ell}^{\min}(0, T_M-\xi, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{a}_C), \tag{3.20}$$

where recursions are carried out in the discrete variables $\mathbf{m}$ and $\mathbf{f}$, the vectors $\mathbf{m}$, $\mathbf{f}$, $\widetilde{\mathbf{a}}_M$, and $\widetilde{\mathbf{a}}_F$, and the scalars 0 and $\widetilde{a}_C$ denote an arbitrarily specified initial system configuration, $\ell = \widetilde{g}(\mathbf{m},\mathbf{f},\mathbf{0})$, $\ell_i = \widetilde{g}(\mathbf{m}-\boldsymbol{\delta}_i,\mathbf{f},\mathbf{0})$, $\ell_i' = \widetilde{g}(\mathbf{m},\mathbf{f}^i,\mathbf{0})$, $\boldsymbol{\delta}_i$ denotes an n-dimensional vector with all its entries equal to zero except that its ith element is equal to one, the vector $\mathbf{v}^{(i)}$ is identical to the vector $\mathbf{v}$ but with its ith component set to zero, and $Q_{\ell}^{\min}(0, T_M-\xi, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{a}_C)$ is the initial condition related to the ℓth integral equation. Moreover, these initial conditions satisfy the system of recursive, coupled integral equations:

$$Q_{\ell'}^{\min}(0,T_M',\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{a}_C) = \int_0^{\infty} \Bigg[\sum_{i=1}^{n} G_{W_{i1}}(\alpha) Q_{\ell_i}^{\min}\Big(0, T_M'-\alpha, (\widetilde{\mathbf{a}}_M+\alpha)^{(i)}, \widetilde{\mathbf{a}}_F+\alpha, \widetilde{a}_C+\alpha\Big) + \sum_{i=1}^{n} G_{Y_i}(\alpha) Q_{\ell_i'}^{\min}\Big(0, T_M'-\alpha, \widetilde{\mathbf{a}}_M+\alpha, (\widetilde{a}_F+\alpha)^{(ii)}, \widetilde{a}_C+\alpha\Big) + G_{\widetilde{Z}^{\min}}(\alpha)\, Q_{\ell''}^{\min}\Big(0, T_M'-\alpha, \widetilde{\mathbf{a}}_M+\alpha, \widetilde{\mathbf{a}}_F+\alpha, \widetilde{a}_C+\alpha\Big)\Bigg] d\alpha, \tag{3.21}$$

where recursions are carried out in the discrete variables $\mathbf{m}$, $\mathbf{f}$, and $\mathbf{c}$, the vectors $\mathbf{m}$, $\mathbf{f}$, $\widetilde{\mathbf{a}}_M$, and $\widetilde{\mathbf{a}}_F$, and the scalars c and $\widetilde{a}_C$ denote an arbitrarily specified initial system configuration, $T_M' = T_M - \xi$, $\ell' = \widetilde{g}(\mathbf{m},\mathbf{f},c)$, $\ell_i = \widetilde{g}(\mathbf{m}-\boldsymbol{\delta}_i,\mathbf{f},c)$, $\ell_i' = \widetilde{g}(\mathbf{m},\mathbf{f}^{(i)},c)$, $\ell'' = g(\mathbf{m}+\mathbf{l},\mathbf{c},0)$, where $\mathbf{l} = (l_1,\ldots,l_n)$ is a column vector describing the total

number of tasks in transit to each server, and the vector $\mathbf{v}^{(i)}$ *is identical to the vector* $\mathbf{v}$ *but with its* i*th component set to zero.*

Corollary 4 (Age-dependent characterization for the lower bound of the service reliability)**.** *Consider an* n*-server DCS whose servers perform a synchronous DTR action at the time* $\xi \geq 0$*. For any* $\ell \in \mathcal{I}$*, the lower bound of the service reliability in executing an application satisfies the system of recursive, coupled integral equations in* ξ*:*

$$R^{\min}_{\ell}(\xi,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{a}_C) = \int_0^{\xi} \Bigg[\sum_{i=1}^{n} G_{W_{i1}}(\alpha) R^{\min}_{\ell_i}\Big(\xi-\alpha,(\widetilde{\mathbf{a}}_M+\alpha)^{(i)},\widetilde{\mathbf{a}}_F+\alpha,\widetilde{a}_C+\alpha\Big) + \sum_{i=1}^{n} G_{Y_i}(\alpha) R^{\min}_{\ell'_i}\Big(\xi-\alpha,\widetilde{\mathbf{a}}_M+\alpha,(\widetilde{\mathbf{a}}_F+\alpha)^{(ii)},\widetilde{a}_C+\alpha\Big)\Bigg] d\alpha + \big(1-F_{\tau_{\mathbf{a}}}(\xi)\big) R^{\min}_{\ell}(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{a}_C), \tag{3.22}$$

where recursions are carried out in the discrete variables $\mathbf{m}$ *and* $\mathbf{f}$*, the vectors* $\mathbf{m}$*,* $\mathbf{f}$*,* $\widetilde{\mathbf{a}}_M$*, and* $\widetilde{\mathbf{a}}_F$*, and the scalars 0 and* $\widetilde{a}_C$ *denote an arbitrarily specified initial system configuration,* $\ell = \widetilde{g}(\mathbf{m},\mathbf{f},\mathbf{0})$*,* $\ell_i = \widetilde{g}(\mathbf{m}-\boldsymbol{\delta}_i,\mathbf{f},\mathbf{0})$*,* $\ell'_i = \widetilde{g}(\mathbf{m},\mathbf{f}^i,\mathbf{0})$*,* $\boldsymbol{\delta}_i$ *denotes an* n*-dimensional vector with all its entries equal to zero except that its* i*th element is equal to one, the vector* $\mathbf{v}^{(i)}$ *is identical to the vector* $\mathbf{v}$ *but with its* i*th component set to zero, and* $R^{\min}_{\ell}(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{a}_C)$ *is the initial condition related to the* ℓ*th integral equation. Moreover, these initial conditions satisfy the system of recursive, coupled integral equations:*

$$R^{\min}_{\ell''}(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{a}_C) = \int_0^{\infty} \Bigg[\sum_{i=1}^{n} G_{W_{i1}}(\alpha) R^{\min}_{\ell_i}\Big(0,(\widetilde{\mathbf{a}}_M+\alpha)^{(i)},\widetilde{\mathbf{a}}_F+\alpha,\widetilde{a}_C+\alpha\Big) + \sum_{i=1}^{n} G_{Y_i}(\alpha) R^{\min}_{\ell'_i}\Big(0,\widetilde{\mathbf{a}}_M+\alpha,(\widetilde{a}_F+\alpha)^{(ii)},\widetilde{a}_C+\alpha\Big) + G_{\widetilde{Z}^{\min}}(\alpha)\, R^{\min}_{\ell''}\Big(0,\widetilde{\mathbf{a}}_M+\alpha,\widetilde{\mathbf{a}}_F+\alpha,\widetilde{a}_C+\alpha\Big)\Bigg] d\alpha, \tag{3.23}$$

where recursions are carried out in the discrete variables $\mathbf{m}$*,* $\mathbf{f}$*, and* $\mathbf{c}$*, the vectors* $\mathbf{m}$*,* $\mathbf{f}$*,* $\widetilde{\mathbf{a}}_M$*, and* $\widetilde{\mathbf{a}}_F$*, and the scalars* c *and* $\widetilde{a}_C$ *denote an arbitrarily specified initial system*

configuration, $T'_M = T_M - \xi$, $\ell' = \widetilde{g}(\mathbf{m}, \mathbf{f}, c)$, $\ell_i = \widetilde{g}(\mathbf{m} - \boldsymbol{\delta}_i, \mathbf{f}, c)$, $\ell'_i = \widetilde{g}(\mathbf{m}, \mathbf{f}^{(i)}, c)$, $\ell'' = g(\mathbf{m} + \mathbf{l}, \mathbf{c}, 0)$, *where* $\mathbf{l} = (l_1, \ldots, l_n)$ *is a column vector describing the total number of tasks in transit to each server, and the vector* $\mathbf{v}^{(i)}$ *is identical to the vector* $\mathbf{v}$ *but with its* i*th component set to zero.*

Proof of Theorems 9 to 12 are sketched in Appendix C since it is a simplified version of the proof of Theorem 5 and 7.

Finally, it must be commented that the approximations and bounds proposed and derived in this section correspond to a simple yet effective dimension reduction technique. The main advantages of this simple approach are that a linear scalability, in the number of servers, has been obtained and that the number of recurrence equations to solve in order to predict the performance and reliability metrics has been dramatically reduced. However, the simplicity of the approach has the disadvantage that no information or knowledge about the process has been exploited to smartly reduce the dimension of the characterization. For instance, one manner of smartly reducing the dimension is to exploit the fact that the conditional probabilities of the regeneration events are know. With this information we can attempt to eliminate all those states with a low likelihood. By implementing such technique in Algorithm 1 we can reduce the number of states visited at earlier recursions, thereby reducing the total number of states visited as part of the evolution of the system dynamics. Another approach that can be taken to reduce the state-space dimension is to investigate the applicability of dimension reductions techniques used in hybrid systems in control theory. For example, the concept of bisimulation is known to be a concept that yields a state space equivalent of a hybrid process, and in addition provides a state space reduction, thereby allowing to study systems with a large state space dimension [14, 87].

3.3 Performance and reliability assessment

3.3.1 Theoretical comparison between age-dependent models and other approaches

At this point, theoretical differences between the Markovian and the non-Markovian characterizations for the performance and reliability metrics are established. First, in a Markovian setting the regeneration time is independent of the age and follows an exponential distribution. In the non-Markovian case the regeneration time follows a general distribution whose parameters are age-dependent. Second, in a Markovian setting the conditional probability associated with the regeneration event remains constant and depends on the parameters of both the random time triggering the regeneration event and the regeneration time. In the non-Markovian case these probabilities are age dependent and depend on the distributions of the random times listed in Assumption A1 as shown in (A.11). Third, in a Markovian setting metrics are characterized by difference-differential equations and algebraic recursions both with having constant coefficients. In the non-Markovian case characterizations are also recursions but comprising integrals with age-dependent coefficients. Fourth, in a Markovian setting the state-space representation for the application execution time is discrete and has a dimension $2n^2 + n$. In the non-Markovian case such state-space has discrete and continuous components and its dimension is $4n^2 + 2n$. Fifth, for a fixed n and a fixed initial state, due to the dimension of the state-vector is smaller in the Markovian case, the number of equations to be solved in such case is smaller than in the non-Markovian case. Figure 3.2 shows the number of states that must be evaluated in the Markovian and the non-Markovian cases as a function of the number of servers. Values were obtained after executing only the first stage of Algorithm 1. Cases labeled as "M fixed" evaluate a situation where an application composed of $M = 400$ tasks is executed by the DCS using n servers. Cases labeled

as "M variable" evaluate a situation where $m_i = 50$ tasks are assigned to each server in the system. (This explains why curves merge for $n = 8$ in each case.) Figure 3.2 shows that the number of states to be evaluated (i.e., the number of integrals to compute) is very large even in the case of two servers. Figure 3.2 also shows that the number of states generated by the non-Markovian representation is about two times the number of states in the Markovian case.

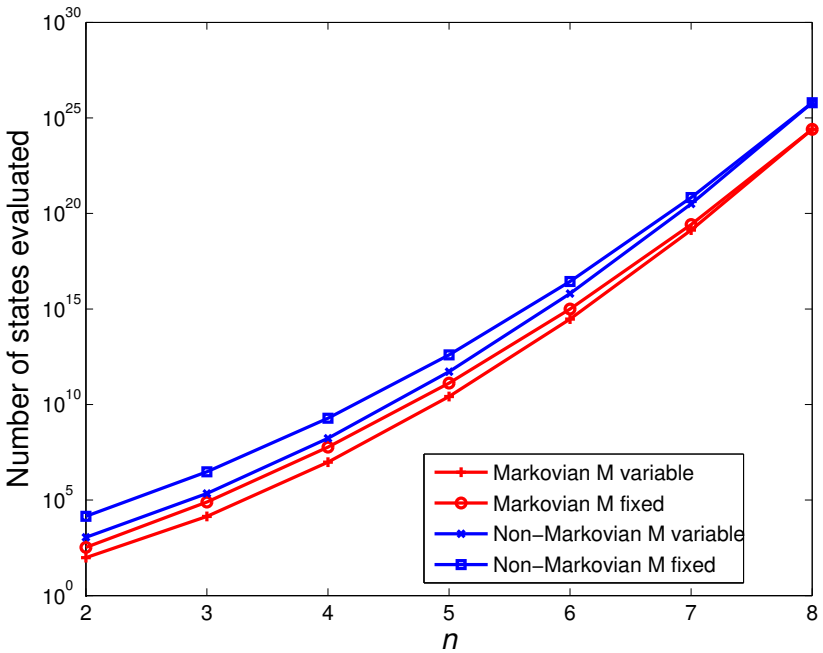


Figure 3.2: Comparison between the number of states generated by the recursions in Theorems 5 to 8 and Corollaries 1 and 2 in Markovian and non-Markovian cases.

Compared to a DES-based performance and reliability analysis, the analytic age-dependent regenerative approach developed here has the following differences and similarities. In terms of memory usage, both the age-dependent state-space model and any DES-based approach must employ a state-space representation of dimension $4n^2 + 2n$ to simulate the class of DCSs regarded here is. Half of these state variables are discrete and must be employed to track the dynamics of queue lengths, failed or working state of servers, and the number of tasks queued in the network. The

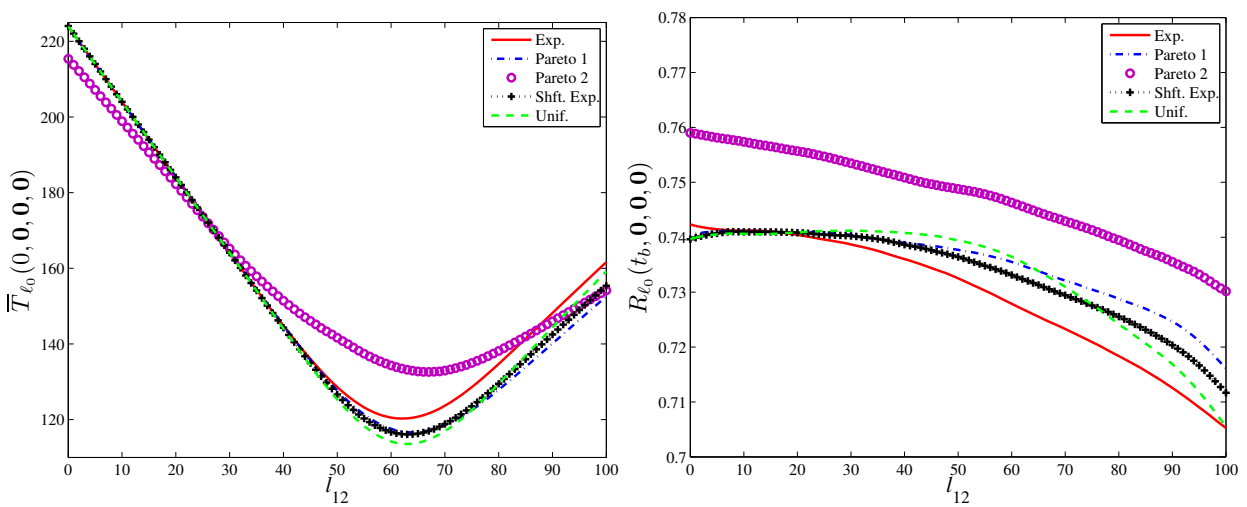
remaining state variables are continuous, however, in DES they must keep track of the remaining time until the next event is triggered in the system, unlike in the age-dependent model where they keep track of the age of the random times. Also in terms of memory usage, the age-dependent model has the disadvantage that a list to track all the states generated at a certain recursion must be stored, and since the list of states is large, so it does the requirement of memory. In terms of computing requirements, the type of calculations performed in a DCS approach are much simpler (mainly random number generation, simple algebraic operations and comparisons) than the CPU-intensive calculations performed to solve the integral recursions given in Theorems 5 to 10. In terms of the number of iterations performed, the age-dependent methods execute a fixed number of iterations, which are given by the total number of states visited as part of the dynamical evolution of the system. In DES, the number of iterations to execute is variable and depends on the confidence required for the results and the use of variance reduction techniques. Moreover, if rare events are being simulated, then an huge number of iterations must be conducted unless techniques like large deviations are employed. Finally, it must be highlighted that the main advantage of the analytical approach taken here over DES is that the results obtained are exact and not approximated results with some associated confidence level, as in the case of simulation.

3.3.2 Comparing Markovian and non-Markovian models

First, predictions for the average execution time, the service reliability and the QoS in executing an application obtained using the non-Markovian characterization to those provided by the Markovian models in [28, 67] will be compared. In the study conducted in this section, the communication network is assumed to be homogeneous and two network-delay conditions have been considered: low and severe network-delays. Under low network-delay conditions, transferring a task to and processing

such task at the fastest server takes, on average, the same time as processing the task at the slowest server. For the severe network-delay case, the average transferring plus processing time of a task to the fastest server is at least five times the service time at the slowest server.

Different stochastic models for the task transfer times have been employed. The Markovian setting is represented by the Exponential model. Pareto 1 and Pareto 2 models represent the case where service and transfer times follow Pareto laws with finite and infinite variance, respectively. For the Shifted-Exponential model, both service and transfer times follow shifted exponential distributions. Finally, in the Uniform model service and transfer times follow uniform distributions. For fair comparison, all distributions modeling the same random times have identical mean values. Without loss of generality, the Markovian and non-Markovian models are compared in a heterogeneous DCSs composed of two servers.



(a) (b)

Figure 3.3: (a) Average execution time; and (b) Service reliability for low network-delay conditions.

The application is composed of 150 tasks, initially allocated as follows: $m_1 = 100$

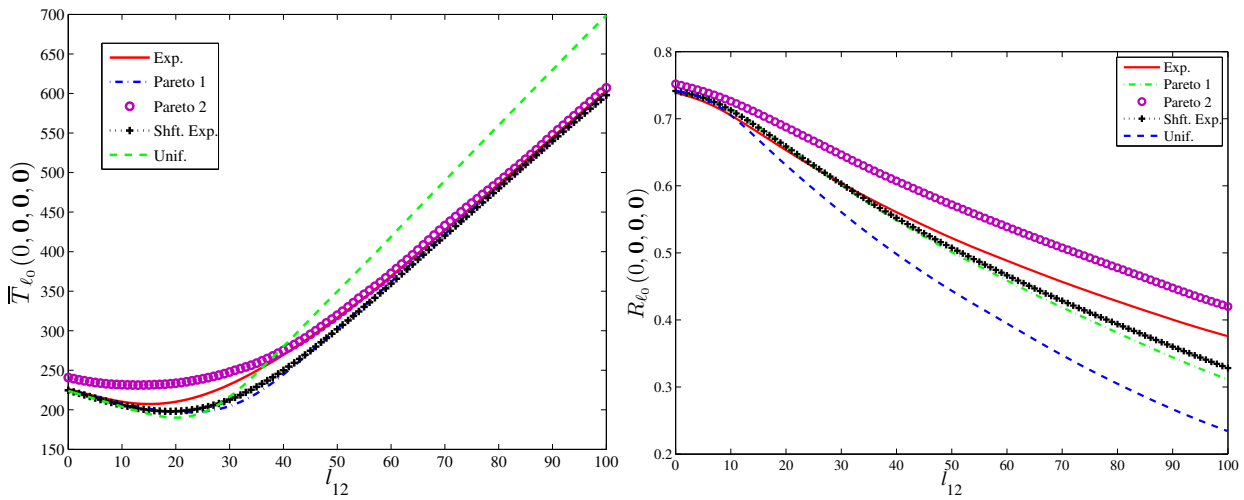


Figure 3.4: (a) Average execution time; and (b) Service reliability for severe network-delay conditions.

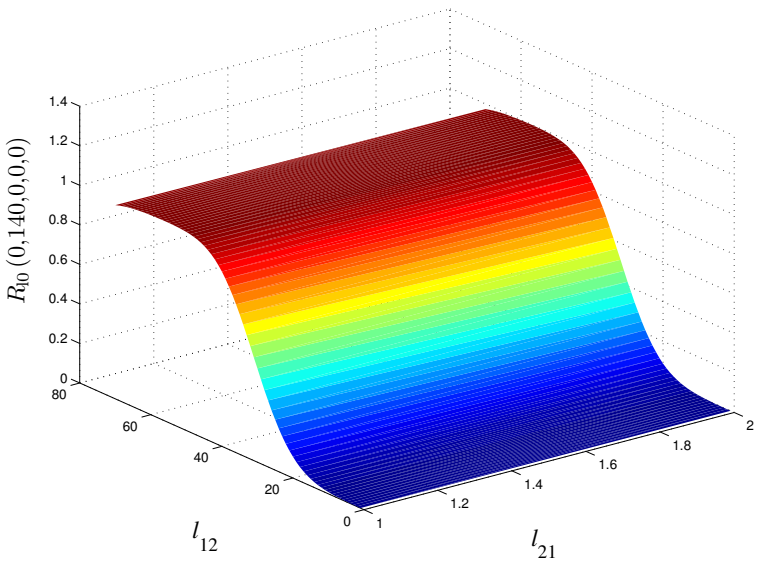


Figure 3.5: QoS in executing an application by the due time $T_M = 140$ s as a function of the number of tasks exchanged.

tasks at server 1 and $m_2 = 50$ tasks at server 2. In order to assess the effect of the network delays on the approximations, it has been assumed in this comparison that

both service and failure times follow exponential distributions. The mean service time per task is 2 and 1 s for servers 1 and 2, respectively, while the mean failure times are $\lambda_{f_1}^{-1} = 1000$ and $\lambda_{f_2}^{-1} = 500$ s. (Recall that when the average execution time is calculated, servers are assumed to be 100% reliable.) Also, the mean transfer time of FN packets are 0.2 and 1.0 s for the low and severe delay scenarios, respectively.

Figures 3.3 and 3.4 show the average execution time and the service reliability as a function of the number of tasks exchanged in the network, for the two network-delay conditions considered. It can be noted that the Markovian approximations for both metrics, the average execution time and the service reliability, show a remarkable accuracy in the low network-delay condition. In fact, the maximum relative approximation errors are below 3% in all cases for both metrics. However, as the mean transfer time increases, Figs. 3.3 and 3.4 also show that the Markovian approximations loose their accuracy in predicting the actual values of the metrics. When network-delays are severe, maximum relative approximation errors increase up to 15% for the average execution time and up to 65% in the case of the service reliability. Figures 3.3 and 3.4 also show the effect of the network-delays on the performance metrics. It can be noted that as the mean transfer time increases so does the average execution time, while the service reliability decreases.

Figure 3.5 shows the QoS metric when the due time is 140 s as a function of the number of tasks exchanged in the network for the Pareto 1 model and severe network-delays. As expected, the QoS in executing the entire application increases as more tasks are transferred from server 1 to server 2. The maximal QoS is only 0.471, which is attained when 33 to 35 tasks are transferred to the second server.

Next, the effect of different stochastic models for the service times on the approximations is assessed. Transfer times as well as failure times have been assumed to follow exponential distributions. The scenario for the transfer times considered is low-network delays. The application served by the DCS has been partitioned into

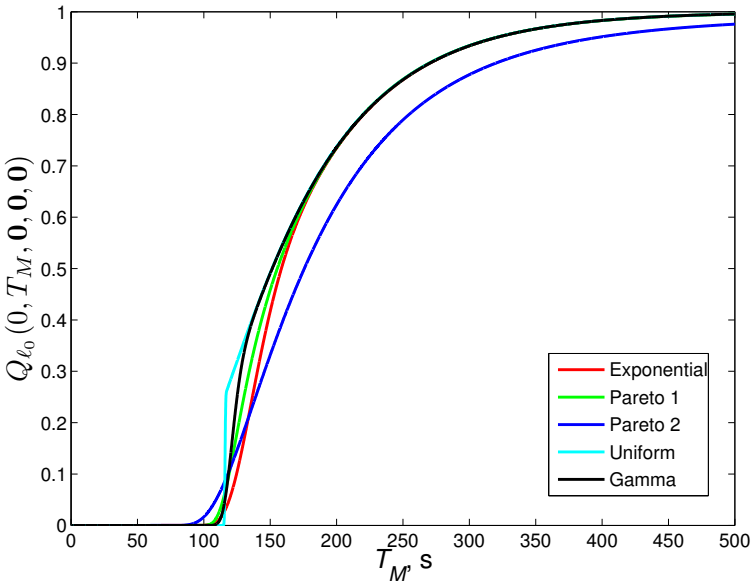


Figure 3.6: The QoS in executing an application as a function of the due time. Several stochastic models and the Markovian approximation are considered for the service time.

$M = 250$ tasks, which are assigned uniformly to a DCS composed of five servers. The average service time of the servers are 1, 2, 3, 4, and 5 seconds. The stochastic models evaluated in this example are Pareto with finite ("Pareto 1") and infinite ("Pareto 2") variance and a uniform distribution with a coefficient of variation (standard deviation divided by the mean) of approximately 2%. As in the previous comparisons all the stochastic models have the same mean. Figure 3.6 compares predictions using the exact stochastic models with the predictions generated by a Markovian approximation ("Exponential") for the QoS metric as a function of the due time . It can be observed from the figure that once again the Markovian approximation yields inaccurate results. In particular, larger approximation errors are obtained in the most extreme cases, that is, in the Pareto 2 model, which has infinite variance, and in the uniform model, which has a very narrow variance and tends to behave as a deterministic model. Moreover, notice that the QoS metric is reduced in the case of

the Pareto 2 model. This is expected as in the Pareto model with infinite variance large service times are more likely to occur as compared to other model with finite variance. This also helps to explain the steep slope in the Uniform model, whose shape and narrow variance eliminates the likelihood of short and long processing times.

3.3.3 Approximations and bounds for the metrics

In order to evaluate both the accuracy of the approximations for the random task transfer times and the tightness of the upper and lower bounds, in this section some theoretical as well as simulation based evaluations are conducted. It must be noted that the simultaneous arrival of tasks to a server at a random time can be defined in several different ways. For instance, a conservative approximation can be obtained by defining $\widetilde{Z}_i$ as the maximum of all the random task-arrival times, that is $\widetilde{Z}_k = \max_i Z_{ik}$. Less conservative approximations can be obtained by defining $\widetilde{Z}_k$ as any ℓth order statistics, where $\widetilde{Z}_k = \min_i Z_{ik}$ corresponds to most aggressive approximation. In the evaluations presented here, the minimum random arrival time as well as the maximum random arrival time approximations have been considered.

Figure 3.7 shows the number of states that must be evaluated for the exact and the approximated characterizations, in the non-Markovian case, as a function of the number of system servers. Once again, cases labeled as "M fixed" evaluate a situation where an application composed of $M = 400$ tasks is executed by the DCS using n servers. Cases labeled as "M variable" evaluate a situation where $m_i = 50$ tasks are assigned to each server in the system. The figure clearly shows the linear dependency of the number of states in the number of system servers, and consequently, shows the large savings in computational resources that the approximation provides.

Results for different initial allocations of an application partitioned in $M = 250$

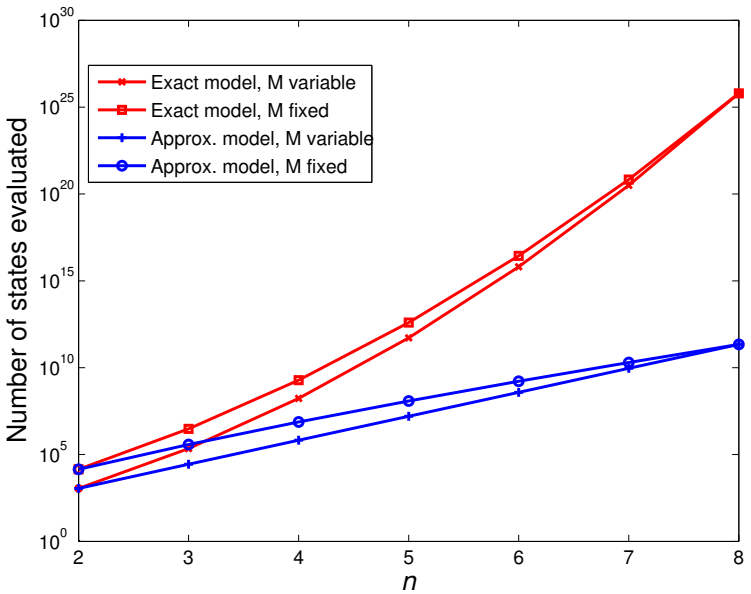


Figure 3.7: Comparison between the number of states generated by the exact and the approximated recursions in Theorems 5 and 7 in the non-Markovian case.

tasks have been calculated. The DCS considered is composed of five servers whose processing times following Exponential or Pareto distributions, with average processing times of 1, 2, 3, 4, and 5 seconds for the first, second, third, fourth, and fifth node, respectively. The communication network has been considered to be homogeneous and the channel parameters are $a_{ij} = a = 3$ (in seconds per task) and $b_{ij} = 1$ (in seconds), $i, j \in \{1, \ldots, 5\}$. In addition, different distributions for the random transfer times of tasks have been considered. Namely, Exponential (Markovian case), Lognormal, Uniform, Gamma and Pareto (with finite and infinite variance). Three different initial allocations for the tasks have been considered: a uniform allocation $m_1 = \ldots = m_5 = 50$ tasks for all i and two non-uniform allocations: $(m_1, \ldots, m_5) = (10, 40, 50, 50, 100)$ and $(m_1, \ldots, m_5) = (150, 0, 0, 0, 100)$. The DTR policy has been executed at time $t_b = 0$ seconds, and it has been assumed that servers possess the same estimate of the queue-length of the remaining servers in the DCS. Moreover, the imbalancing in the DCS is detected based on the relative processing

speed of the nodes. From this, it can be calculated that the total amount of tasks exchanged over the network is 61, 111, and 117 tasks when the initial allocation is (50,50,50,50) and (10,40,50,50,100), and (150,0,0,0,100), respectively.

Figures 3.8 and 3.9 show results for the case when the random service times follow exponential distributions. MC simulations are also shown representing the exact value of the QoS. The QoS metric estimated via MC simulation was obtained using a 95% confidence. The first observation that can be made about the approximations is that they are not very accurate, and that the bounds are not tight. The second observation is that the approximation based on the maximum random arrival time per server is more accurate than the aggressive approximation based on the minimum arrival time. It can be observed from the figures that as the number of tasks exchanged over the network increases so it does the execution time of the application. In addition, it can be observed that the execution time of the application becomes excessively large when the random communication times follow Pareto and Lognormal distributions. This is attributed to the fact that such distributions possess a "heavy tail," which means that events taking a long time to occur are more likely, as compared to distributions not exhibiting a heavy tail such as the exponential distribution. Figure 3.10 shows the results obtained when the random service times follow Pareto distributions with a finite variance for the non-uniform case (150,0,0,0,100). It can be noted that the execution times of the application are longer than in the case of exponentially distributed service times.

3.4 Conclusions

A novel analytical characterization for the average execution time, the service reliability and the QoS in executing an application in DCSs has been developed. The age-dependent analytical characterizations constitute a generalization to a non-

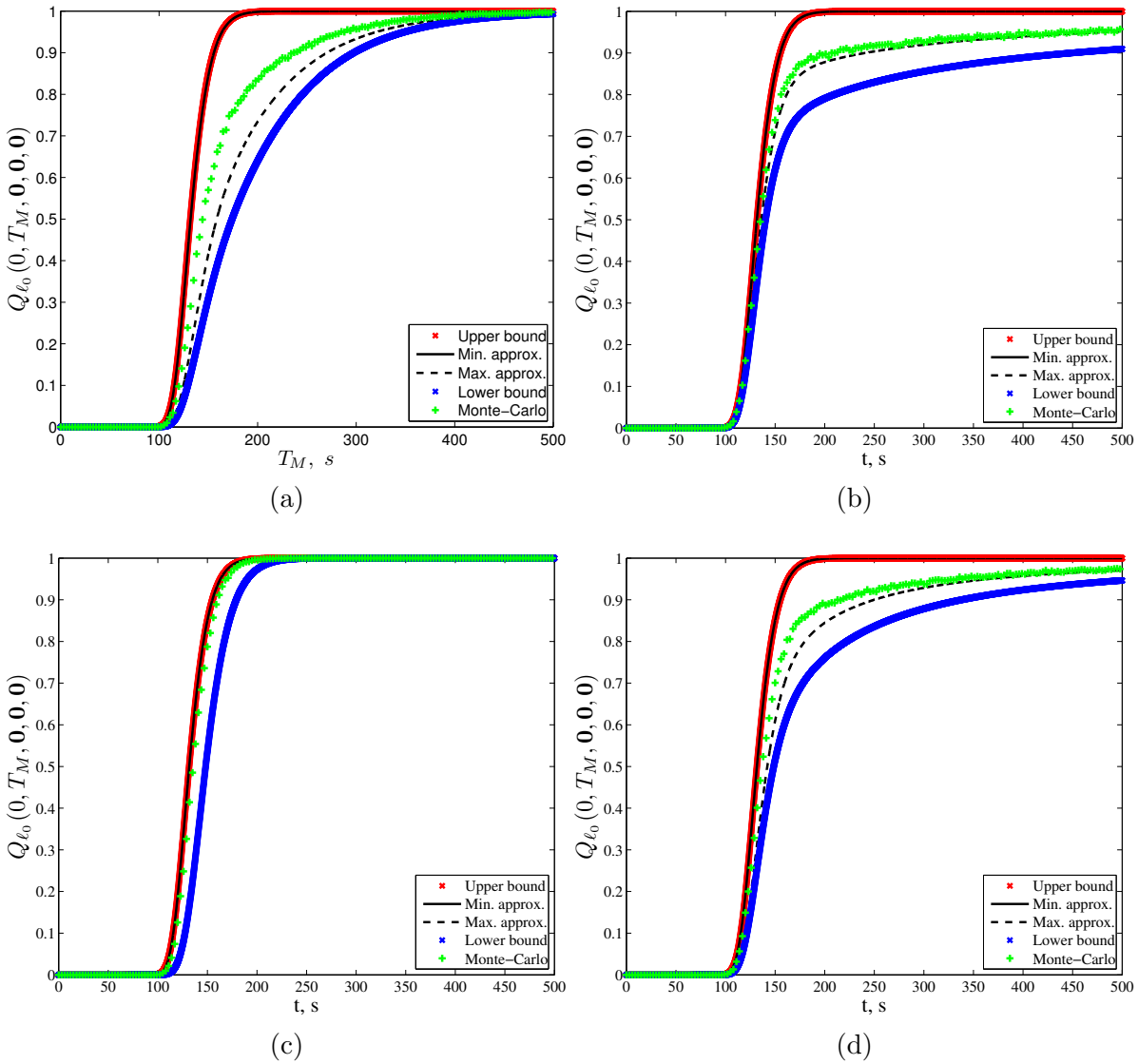


Figure 3.8: Approximated QoS using the minimum and the maximum arrival time of tasks reallocated for the balanced case. Random service times follow exponential distributions. Random transfer times of tasks follow: (a) Exponential, (b) Lognormal, (c) Gamma, and (d) Pareto (with infinite variance) distributions.

Markovian, and hence, more physical setting of the work on LB in DCSs conducted by Dhakal in [28]. The analysis of these performance and reliability metrics in this new setting is important since they provide insights on the accuracy of the expo-

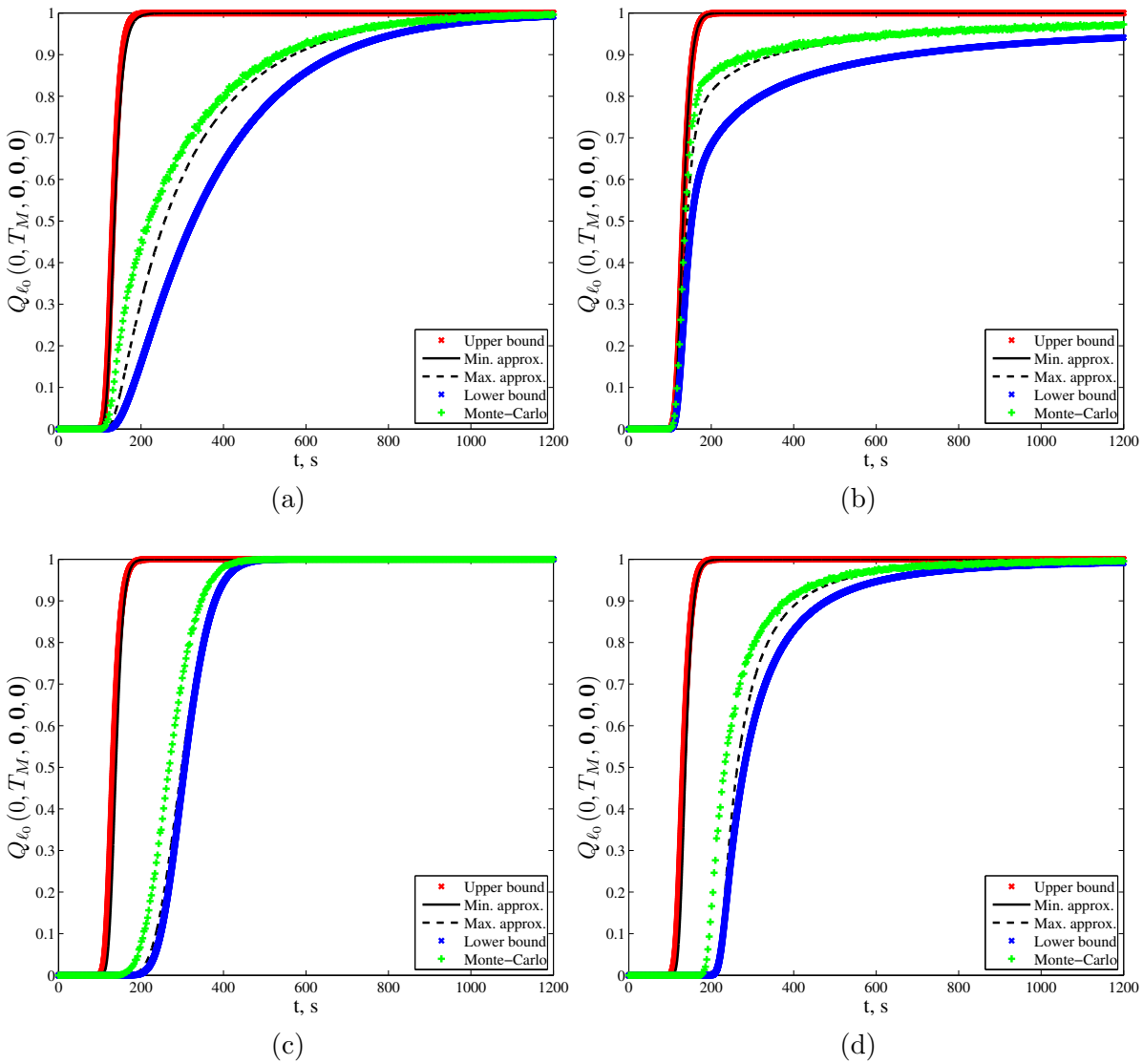


Figure 3.9: Approximated QoS using the minimum and the maximum arrival time of tasks reallocated for the unbalanced case $(m_1, \ldots, m_5) = (10, 40, 50, 50, 100)$. Random service times follow exponential distributions. Random transfer times of tasks follow: (a) Exponential, (b) Lognormal, (c) Gamma, and (d) Pareto (with infinite variance) distributions.

nential approximation in modeling actual phenomena that not follow exponential approximations. Results obtained here have shown that when the variance of the

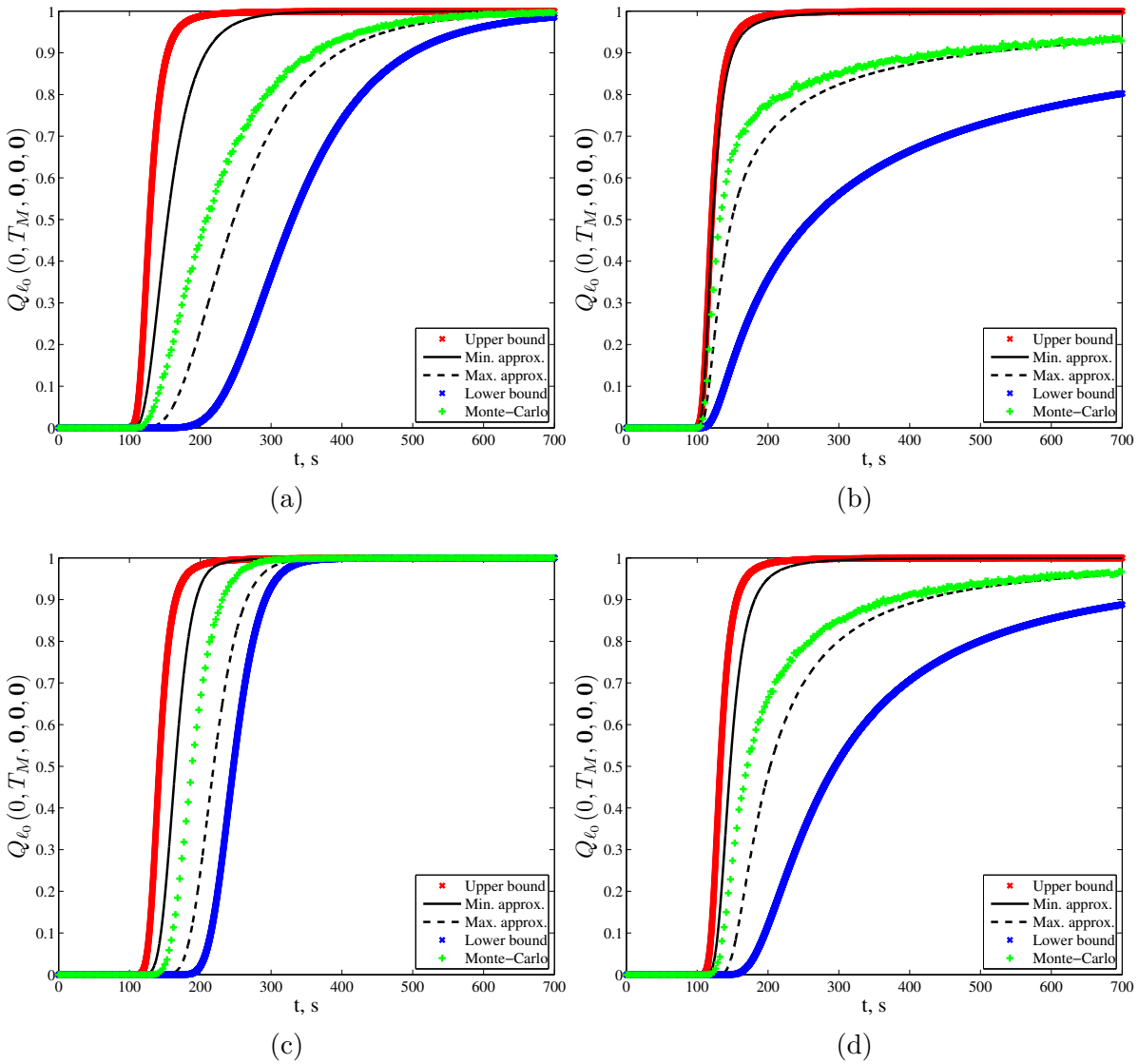


Figure 3.10: Approximated QoS using the minimum and the maximum arrival time of tasks reallocated for the unbalanced case $(m_1, \ldots, m_5) = (150, 0, 0, 0, 100)$. Random service times follow Pareto distributions with finite variance. Random transfer times of tasks follow: (a) Exponential, (b) Lognormal, (c) Gamma, (e) Pareto (with infinite variance) distributions.

random times in the DCS is large, Markovian approximations produce inaccurate results, which can be significant when approximating probabilities like in the case

of QoS and reliability metrics. Moreover, this generalization has provided also insights about the effect of network-delays on the service time of applications and their effects on the modeled performance and reliability metrics. Results provided here indicate that when the network delays are relatively large compared to service times, the error in estimating any of these metrics, as a result of falsely assuming exponentially distributed random delays, becomes significant, thereby necessitating the use of the age-dependent model developed in this dissertation. For example, our calculations show relative errors as large as 40% and 110% in estimating the average execution time and the service reliability, respectively. It must be mentioned that the accuracy provided by the age-dependent model in predicting the three performance metrics regarded here comes at expense of increased computations as compared to their Markovian counterpart.

The stochastic model for DCS developed here takes into account the heterogeneity in the computing resources, the stochastic communication, the uncertainty associated with the number of functional servers in the DCS, and an arbitrary DTR policy executed by the servers. The models is based on a hybrid age-dependent state vector that tracks the discrete variables of the system (number of tasks queued at the servers and in the network as well as the number of functioning servers) and the memory associated with the non-exponential random times. This state vector effectively tracks the underlying point processes associated with the dynamics of the DCS. The mathematical framework presented here can be modified to calculate other performance metrics, such as statistics of the queue-length of servers or jitter in the service time, and can be extended to model crash-recovery failure scenarios. The latter extension, however, has the extra complication that increases the number of states visited by the process, and consequently, the computational burden is largely augmented.

Mathematical approximations and bounds for the metrics characterized here have

been developed to reduce the computational resources required to calculate the exact characterizations. The analytical approximations presented significatively reduce the number of calculations; however, their accuracy in the prediction of the metrics may not be acceptable. This has been observed in situations when several servers are exchanging tasks over the network. Finally, it must be commented that the analytical bounds presented here are not tight.

Chapter 4

Optimal task reallocation in distributed computing systems

The problem of efficiently executing a parallel application on a DCS is a fundamental problem in distributed computing. Solutions to this problem aim to devise smart DTR strategies that judiciously employ the resources of the DCS. These strategies are run-time control actions dictating when and how tasks must be exchanged among the servers. In a DCS, effective DTR strategies must account for the heterogeneous processing capacities of the servers and the heterogeneous transfer delays imposed by the communication network. The problem becomes challenging when DC is performed in harsh scenarios where servers may fail permanently, because effective DTR policies must consider also the reliability of the servers, while accurate estimates of the working or failed state of servers are either not available or dated.

4.1 Problem statement

Given a DCS composed of n servers and a parallel application partitioned into M tasks, the DTR problem consists in finding the optimal task migration scheme such that the metrics characterized in Theorems 5 to 8 are optimized. In this dissertation the DTR problems regarded are minimizing the average service time, and maximizing both the service reliability and the QoS in serving an application. Thus, the following mixed integer optimization problems are formulated.

Minimal average execution time:

$$(t_b^*, \mathbf{L}^*) = \underset{(t_b,\mathbf{L})}{\operatorname{argmin}} \ \overline{T}_{\ell_0}(t_b, \mathbf{0}, \mathbf{0}, \mathbf{0}), \tag{4.1}$$

subject to:

$$\sum_{j=1, j\neq i}^{n} l_{ij} \leq m_i, \ i = 1, \ldots, n, \tag{4.2}$$

$$l_{ij} \in \{0, 1, \ldots, m_i\}, \ i, j = 1, \ldots, n, \ i \neq j, \tag{4.3}$$

$$t_b \geq 0. \tag{4.4}$$

Maximal quality-of-service:

$$(t_b^*, \mathbf{L}^*) = \underset{(t_b,\mathbf{L})}{\operatorname{argmax}} \ Q_{\ell_0}(t_b, T_M, \mathbf{0}, \mathbf{0}, \mathbf{0}), \tag{4.5}$$

subject to:

$$\sum_{j=1, j\neq i}^{n} l_{ij} \leq m_i, \ i = 1, \ldots, n, \tag{4.6}$$

$$l_{ij} \in \{0, 1, \ldots, m_i\}, \ i, j = 1, \ldots, n, \ i \neq j \tag{4.7}$$

$$t_b \geq 0. \tag{4.8}$$

Maximal service reliability:

$$(t_b^*, \mathbf{L}^*) = \underset{(t_b,\mathbf{L})}{\operatorname{argmax}} \; R_{\ell_0}(t_b, \mathbf{0}, \mathbf{0}, \mathbf{0}), \tag{4.9}$$

subject to:

$$\sum_{j=1, j\neq i}^{n} l_{ij} \leq m_i, \; i = 1, \ldots, n, \tag{4.10}$$

$$l_{ij} \in \{0, 1, \ldots, m_i\}, \; i, j = 1, \ldots, n, \; i \neq j \tag{4.11}$$

$$t_b \geq 0. \tag{4.12}$$

Each problem has $n(n-1)$ non-negative integer-valued variables, one non-negative real-valued variable and $n^2 + 1$ restrictions.

From (4.1) and (4.5) it becomes clear that the optimization problem is mixed integer. Since the optimization problem is known to be NP-hard due to the combinatorial explosion of the search space, an efficient search algorithm must be devised in order to find feasible optimal DTR policies. Before discussing the search algorithm, the general DTR policy employed by the servers is introduced.

4.1.1 Distributed task reallocation policy

In this dissertation, the following DTR policy, consisting in three steps, executed in a synchronous and distributed manner by the servers in DCS is considered.

Step 1. Since the DTR policy executed by the servers is distributed, each server must determine independently the total amount of tasks to reallocate to other servers. At the prescribed instant, $t_b \geq 0$, the jth functioning server computes its excess load by comparing its local load to the estimated average load in the system. Recall that $m_j(t_b)$ denotes the number of tasks queued at the jth server at time t_b. Let $\hat{m}_{\ell,j}(t_b)$ be the estimate of the number of tasks queued at the ℓth functioning server

as perceived by the jth server at time t_b, with $\ell \neq j$. The excess load of the jth server at time t_b is defined as

$$L_j^{ex}(t_b) \triangleq m_j(t_b) - \frac{\Lambda_j}{\sum_{\ell \in \mathcal{W}_j} \Lambda_\ell} \hat{M}_j(t_b), \tag{4.13}$$

where $\hat{M}_j(t_b) = m_j(t_b) + \sum_{\ell=1,\ell\neq j}^{n} \hat{m}_{\ell,j}(t_b)$ is the estimate of the number of tasks in the system as perceived by the jth server at time $t = t_b$, $\mathcal{W}_j$ is the collection of servers that are functioning as perceived by the jth server at time $t = t_b$, and the Λ_j's are parameters that can be defined in several ways in order to establish different reallocation criteria.

For example, the Λ_j parameters may be associated with the processing speed of the servers. If the processing speed of the jth server is denoted as $\lambda_j = 1/\mathsf{E}[W_{j1}]$, then Λ_j are given by $\lambda_j = 1/\mathsf{E}[W_{j1}]$, and the imbalance in the DCS is determined by the relative computing power of the servers. Alternatively, the Λ_j's may be associated with the reliability of the servers, λ_j^f, which is defined as $\lambda_j^f = 1/\mathsf{E}[Y_j]$. In this case, the relative resilience of the servers determines the amount of imbalance in the DCS. Yet another option is to define the Λ_j's so that, simultaneously, a server transfers fewer tasks to the less-reliable servers and transfers larger number of tasks to the faster servers. With this criterion in mind, the Λ_j parameters can be defined as

$$\Lambda_j = \lambda_j \Big(1 - \frac{\lambda_j^f}{\sum_{k \in \mathcal{W}_j} \lambda_k^f}\Big). \tag{4.14}$$

Note that in the case of an extremely reliable server, ($\lambda_i^f \approx 0$), the parameter Λ_j is approximately equal to the processing rate of a server. On the contrary, for an unreliable server the parameter Λ_j is only a reduced fraction of its processing rate.

Step 2. Each server in the DCS has to determine the amount of tasks to reallocate to the remaining servers in the system. Let $\mathcal{V}$ denote the collection of overloaded servers in the DCS. This collection is defined as all those servers that, at the reallocation instant, perceive themselves as overloaded with respect to their perceived

fair share of the total workload of the system. More precisely, $\mathcal{V} \triangleq \{j : L_j^{ex}(t_b) > 0\}$. Similarly, for each overloaded server j, the collection $\mathcal{U}_j$ of candidate task-receiver servers is defined as all those servers that, at time t_b, are perceived by the sender server as functioning and underloaded with respect to their own perceived fair shares of the total workload; namely, $\mathcal{U}_j \triangleq \{k : L_{k,j}^{ex}(t_b) < 0,\ k \in \mathcal{W}_j\}$, where $j \in \mathcal{V}$ and $L_{k,j}^{ex}(t_b)$ is the excess load at the kth functioning server as perceived by the jth server and is defined as $L_{k,j}^{ex}(t_b) \triangleq \hat{m}_{k,j}(t_b) - \Lambda_k \hat{M}_j(t_b) / \sum_{\ell \in \mathcal{W}_j} \Lambda_\ell$.

Step 3. The jth server partitions its excess load among all the candidate task-receiver servers. For the kth candidate task-receiver server, the partition p_{jk} is defined as $p_{jk} \triangleq L_{k,j}^{ex}(t_b) / \sum_{\ell \in \mathcal{U}_j} L_{\ell,j}^{ex}(t_b)$ whenever $k \in \mathcal{U}_j$. For convenience, the partition $p_{ji} = 0$ for all $i \notin \mathcal{U}_j$. In general, the load partitions p_{jk} may not be effective and must be adjusted in order to compensate for the effects of the random transfer times. This compensation is carried out by the algorithm to be presented next. However, the unadjusted partitions will be used to obtain:

$$l_{ij}^{(0)} \equiv l_{ij}^{(0)}(t_b) = \left\lfloor p_{ij} \Big(m_i(t_b) - \frac{\Lambda_j}{\sum_{\ell \in \mathcal{W}_j} \Lambda_\ell} \hat{M}_i(t_b) \Big) \right\rfloor, \tag{4.15}$$

which will be used as initial values to start the search algorithm described next.

Scalable algorithm for sub-optimal task reallocation

For DCSs with arbitrary number of servers, it has already been shown that it is computationally expensive to calculate the performance and reliability metrics using the exact characterizations. In addition, the optimization problem is NP-hard and demands for also fast search algorithms. To circumvent all these problems, an efficient iterative algorithm for devising sub-optimal DTR policies is provided. The idea of the algorithm is to exploit the decomposition technique used in mixed integer programming, and solve the optimization problems (4.1) and (4.5) for two-server DCSs. Key observations in the development of this algorithm are that the exact

characterization of the reliability metrics for two-server systems are computationally inexpensive, and that such pairwise decomposition yields a linear scalability in the number of system servers. This algorithm was reported in [67] and refined in [66,70].

The algorithm computes the number of tasks to reallocate from the ith to the jth server at the kth iteration, $l_{ij}^{(k)}$, as follows. The ith server has an estimate, $\hat{m}_{j,i}$, of the number of tasks queued at the jth server. Using these estimates, the ith server constructs its collection of candidate task-recipient servers, U_i. From such collection, the ith server picks the jth server, say, and obtains $l_{ij}^{(k)}$ by solving either (4.1) or (4.5) with $m_1 = r_i$ and $m_2 = \hat{m}_{j,i}$, where r_i is the number of tasks queued at the ith server assuming that such server has already reallocated tasks to all its other candidate recipient servers, with the exception of the jth server. In order to produce an algorithm independent of the order in which servers are selected from U_i, the $l_{ij}^{(k)}$ quantities are iteratively computed until all of them settle down to some value or until a maximum number of iterations, K, is reached. The algorithm requires the following parameters: K, $\hat{m}_{j,i}$, t_b and $l_{ij}^{(0)}(t_b)$. The parameter K is selected by the user. The estimates $\hat{m}_{j,i}$ are obtained from queue-length information packets frequently exchanged among the servers, and the initial value for the DTR policy, $l_{ij}^{(0)}$, is given by (4.15).

A pseudo code for the algorithm is shown in Algorithm 2. The scalability of Algorithm 2 has been studied empirically. As the algorithm is decentralized, the study focuses only on the computations performed by a single server. The most computationally intensive case has been considered in this study. That is, when a server has to partition its excess load among all the other servers in the DCS. Figure 4.1 shows the average computing overhead introduced by Algorithm 2, i.e., the execution time taken by the multi-server algorithm to solve (4.1), as a function of the number of servers in the DCS, for four different amounts of tasks allocated at the sender server. The mean computing overhead was computed by averaging 100

Algorithm 2 DTR policy for multi-server DCSs

Require: K, $\hat{m}_{j,i}$ and $l_{ij}^{(0)}$, with $j = 1, \ldots, n$, $i \neq j$

Ensure: $\mathbf{L} = (l_{ij})_{n \times n}$

Set $U_i = \{j : L_{ij}^{(0)} > 0\}$, $U_i^{'} = \emptyset$ and $k = 1$

loop

while $j \in U_i$ **do**

$U_i \leftarrow U_i \setminus \{j\}$

$m_1 = m_i - \sum_{\ell \in U_i} L_{i\ell}^{(k-1)} - \sum_{\ell \in U_i^{'}} l_{i\ell}^{(k)}$ and $m_2 = \hat{m}_{j,i}$

Solve (4.1) or (4.5) using m_1 and m_2 to obtain $l_{ij}^{(k)}$

$U_i^{'} \leftarrow U_i^{'} \cup \{j\}$

end while

Set $U_i = \{j : L_{ij}^{(0)} > 0\}$, $U_i^{'} = \emptyset$ and $k \leftarrow k + 1$

if $\sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} \left(l_{ij}^{(k)} - l_{ij}^{(k-1)}\right) = 0$ or $k > K$ **then**

$l_{ij} = l_{ij}^{(k)}$ for all $j \in U_i$ and **exit**

end if

end loop

executions of the algorithm. Roughly speaking, Fig. 4.1 shows that the overhead increases linearly with the number of servers in the DCS. Moreover, note that each server has to solve at most $n - 1$ times any of the two optimization problems, and such computation has to be repeated no more than K times. From this, it can be observed that the algorithm in fact scales linearly in the number of servers.

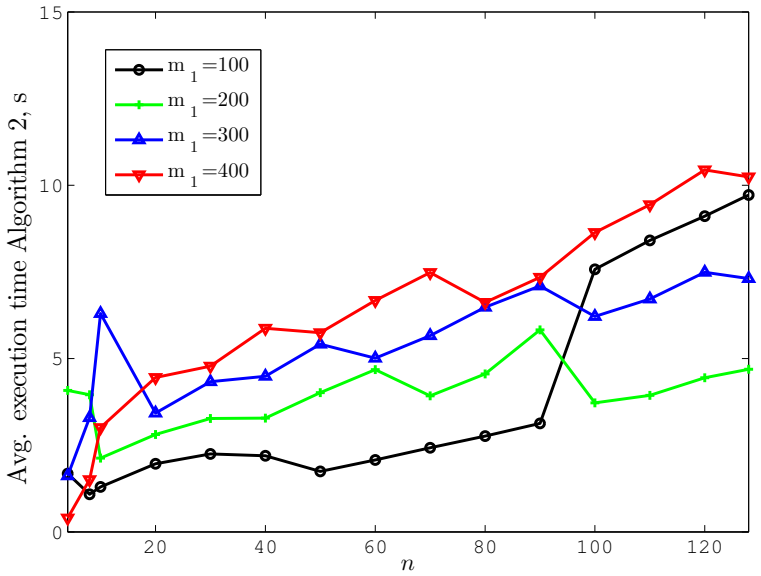


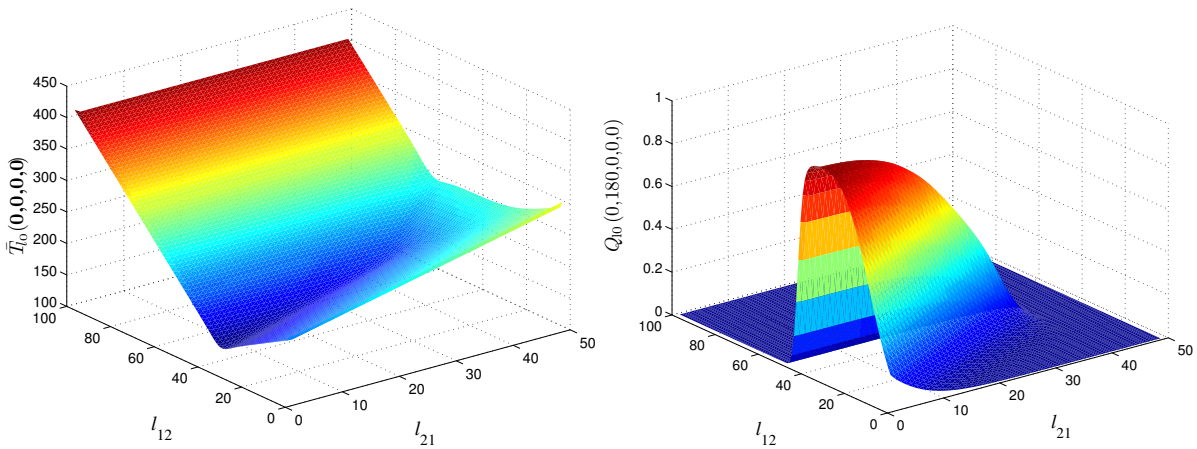
Figure 4.1: Average execution time taken by Algorithm 2 to solve (4.1) as a function of the number of system servers.

4.2 Task reallocation in non-Markovian settings

4.2.1 Task reallocation for two-server systems

Since Algorithm 2 is based upon a two-server decomposition of a DCS, results for two-server DCSs are presented first. As in Section 3.3.2, the parallel application to be executed on the DCS comprises $m_1 = 100$ tasks and $m_2 = 50$ tasks, initially allocated at servers 1 and 2, respectively. The mean service time per task is 2 and 1 s for servers 1 and 2, respectively. It has been also assumed that failure times follow exponential distributions with means $\lambda_{f_1}^{-1} = 1000$ and $\lambda_{f_2}^{-1} = 500$ s. (Recall that when the average service time is calculated, servers are assumed to be 100% reliable.) Also, the mean transfer time of FN packets are 0.2 and 1.0 s for the low and severe network delay scenarios, respectively.

The optimization problems (4.1) and (4.5) are solved in order to devise DTR



(a) (b)

Figure 4.2: (a) Average service time, and (b) QoS in executing an application before $T_M = 180$ s, as a function of all the DTR policies for Pareto 1 model and severe network-delay conditions.

policies for the average service time and the QoS in serving an application. For both performance metrics the case of a DCS with completely reliable servers has been considered. Optimization results for all the models and both low and severe network-delay conditions are listed in Table 4.1. It can be seen from such Table that when network-delays are low, a Markovian model yields fairly accurate predictions for both metrics. However, when the amount of delay in the communication network is high, results produced by the Markovian approximation are not only inaccurate, but also yield DTR policies that can degrade the performance metrics in approximately 10 to 40%. Figure 4.2 shows the average service time and the QoS in serving an application as a function of the DTR policies for the Pareto 1 model and severe network-delays. The minimal average service time is 140.11 s, and is achieved by the policy $l_{12} = 32$ tasks and $l_{21} = 1$ tasks. Figure 4.2(b) shows the QoS in serving the entire application within 180 s. The QoS metric for $T_M = 180$ s is maximized by three policies $l_{12} \in \{31, 32, 33\}$ tasks and $l_{21} = 1$ task, which yield a maximal probability of serving the application of 0.988. It must be commented that the maximal QoS in

serving the application within 140 s (the minimal average completion time) is only 0.471.

Next, the effect of optimal DTR policies on the usage of the computing resources is discussed. If low network-delay conditions are considered, optimal policies dictate that approximately 50% of the load initially allocated at the slower server has to be migrated to the faster server, while the latter server must keep all its initial load. Note that, on average, server 2 processes its initial load in 50 s, and note also that, transferring 50 tasks from server 1 to server 2 takes 50 s. Consequently, the optimal task reallocation is perceived by the second server as an instantaneous exchange of load. In addition, note that processing the 50 tasks reallocated to server 2 takes another 50 s, on average, while serving the remaining 50 tasks at server one takes 100 s, on average. Therefore, optimal policies keeps both servers busy for approximately the same amount of time, thereby efficiently using the computing resources of the DCS. When network-delays are severe, computing resources cannot be utilized equally. In this case, optimal policies trade off between transfer times and utilization of the servers.

4.2.2 Task reallocation for multi-server systems

The average service time and the service reliability of a heterogeneous, five-server DCS have also been optimized by employing Algorithm 2 presented in Section 4.1.1. It has been assumed that the parallel application is partitioned in $M = 200$ tasks. To assess the service reliability, it has been assumed that failure times follow Weibull distributions with means 1000, 800, 600, 500, and 400 s, for servers 1 to 5, respectively. The average service times were set to be 5, 4, 3, 2, and 1 s for servers 1 to 5, respectively. The remaining parameters are the same as those in the two-server analysis.

Table 4.1: Optimal DTR policies for average service time and QoS.

	Low network-delays			
Model	$\overline{T}_{\ell_0}(0,\mathbf{0},\mathbf{0},\mathbf{0})$	l_{12}	$Q_{\ell_0}(0,110,\mathbf{0},\mathbf{0},\mathbf{0})$	l_{12}
Exponential	110.65	51	0.604	53
Pareto 1	106.44	53	0.660	53
Pareto 2	106.26	53	0.675	53
Shifted Exp.	108.81	52	0.642	52
Uniform	112.33	54	0.630	54
	Severe network-delays			
Model	$\overline{T}_{\ell_0}(0,\mathbf{0},\mathbf{0},\mathbf{0})$	l_{12}	$Q_{\ell_0}(0,180,\mathbf{0},\mathbf{0},\mathbf{0})$	l_{12}
Exponential	183.80	16	0.659	22
Pareto 1	140.11	32	0.988	33
Pareto 2	119.10	44	0.997	44
Shifted Exp.	139.94	40	0.969	39
Uniform	145.32	37	0.979	37

Table 4.2: Service reliability for different models under severe network-delay conditions.

Initial load	Avg. Service Time, s				
$(m_1,\ldots,m_5)$	Pareto 1	Pareto 2	Shft. Exp.	Uniform	Exp.
(200,0,0,0,0)	109.66	101.33	108.41	106.40	128.62
(0,0,0,0,200)	107.39	102.62	107.01	105.55	125.57
(40,40,40,40,40)	105.44	101.91	104.31	104.79	122.83
(18,22,30,43,87)	90.01	88.87	89.37	91.47	90.05
Initial load	Service Reliability				
$(m_1,\ldots,m_5)$	Pareto 1	Pareto 2	Shft. Exp.	Uniform	Exp.
(200,0,0,0,0)	0.614	0.622	0.601	0.572	0.339
(0,0,0,0,200)	0.588	0.591	0.575	0.601	0.392
(40,40,40,40,40)	0.631	0.643	0.632	0.639	0.430
(62,48,36,30,24)	0.741	0.755	0.729	0.699	0.546

Table 4.2 lists both the average service time and the maximal service reliability obtained under severe network-delay conditions. Both performance metrics were obtained by solving the system of equations generated by the recursion presented in Theorem 5. For comparison, the column "Exponential" presents results yielded using the optimal policies devised under Markovian assumptions. Also for comparison, the last row on each part of the Table represents a benchmark for each performance

metric, since the initial allocation of tasks is actually the optimal allocation. These optimal task allocations were obtained by performing a MC-based exhaustive search over all the DTR policies. It can be noted from Table 4.2 that the exponential approximation produces relative errors between 5% and 45%. As in the case of two-server DCSs, using incorrect models for the random times yield not only inaccurate results, but also specifies inappropriate reallocation policies that, in turn, reduce the performance metrics under study. Finally, during the evaluations conducted in this work it has been observed that policies devised using Algorithm 1 and a non-Markovian model can achieve values for the average service times and the service reliability that are within 70% of the optimal values.

4.2.3 Distributed computing testbed

As part of this dissertation, the DCS architecture developed by Ghanem in [40] has been employed and modified to accommodate independent as well as correlated failures. The hardware architecture consists of the computing servers and the communication network. The set of computing servers comprises heterogeneous processors, such as Pentium II- and Pentium 4-based computers. Lately, the testbed has been ported to a dedicated cluster of twelve PowerPC-based computers. This cluster belongs to the UNM Center for Advanced Research Computing [86]. Depending on which environment is used some of the computing servers are dedicated machines (cluster), while others are serving as lightly loaded web-, mail- and database-servers. The occurrence of failures at the server is simulated by software. In the case of independent failures, each server randomly generates a time to fail. In the case of correlated failures, the server in charge of providing the initial allocation of load and triggering the DC draws a failure pattern using Algorithm 3 and generates a random failure time. The failure pattern as well as the random failure time are broadcasted to the remaining servers in the system using the same data packet employed to trig-

ger the beginning of the execution of the application. These methods for generating and simulating independent and correlated failures are practical contributions of this work to enhance the testbed architecture. Upon the occurrence of a failure, a computing server is switched from the so-called working state to the failed state. If a server is in the failed state then it cannot process tasks. The communication network employed by the testbed architecture is mainly the Internet, where the final links connecting the computing nodes are either wired or wireless. Hence, the communication network naturally exhibits a notorious communication delay. However, when the cluster of PowerPC-based servers is employed or in situations where some of the communication links provide high speed connection, artificial latency has been introduced to the communication by means of traffic shaper applications. A traffic shaper may reduce the actual transfer speed of the network interfaces to slow speeds such as 1024 to 512 Kbps. This is the second practical contribution of this work to the testbed DCS.

The software architecture of the DCS is divided in three layers: application, task reallocation and communication. Layers are implemented in software using POSIX threads. The application layer executes the application selected to illustrate the DC: matrix multiplication. The service of a task has been defined as the multiplication of one row by a static matrix, which is duplicated in all servers. To achieve variability in the processing time of the servers, randomness is introduced in the size of each row by independently choosing its arithmetic precision using any arbitrary probability distribution. The application layer also switches the state of a server from working to failed. The same layer maintains, at each server, two vectors of $n-1$ components that track the failed or working state of the other servers in the DCS. The first vector stores the number of tasks queued at the other nodes using a long integer representation. The second vector is binary and indicates which servers remain functioning in the system. The task reallocation layer executes the DTR policy defined for each type of experiment conducted. This layer schedules and triggers the

reallocation instants when task-reallocation is performed. It also: (i) determines if a server is overloaded with respect to the other servers in the system; (ii) selects which servers are candidate receiving servers; and (iii) computes the amount of task to transmit to the receiver nodes by using Algorithm 2. Finally, the communication layer of each server handles the transfer of tasks as well as the transfer of FN packets among the servers. Each node uses the UDP transport protocol to transfer either an FN packet to the other servers. The TCP transport protocol is used to transfer tasks between the servers.

4.2.4 Maximizing the service reliability of a testbed DCS

In order to experimentally validate the age-dependent theory developed here, policies for maximizing the service reliability of a testbed DCS, which uses the Internet as the communication network, have been devised. A detailed description of the testbed DCS employed is given in [67]. In order to yield predictions for the service reliability, first it is mandatory to characterize experimentally the random times of the testbed DCS. Figures 4.3(a) and (b) show the normalized histograms as well as fitted pdfs for the service time of server 1 and the transfer time of tasks from server 2 to 1. The parameters of the fitted pdfs were estimated using maximum likelihood estimators. Each estimated pdf was selected according to the minimum total squared error between the normalized histogram and each fitted pdfs. From this experimental characterization, it has been found that: (i) the service times at servers 1 and 2 follow Pareto distributions with means 4.858 and 2.357 s, respectively; (ii) the task transfer times follow shifted gamma distributions with means $\overline{Z}_{12} = 1.207$ and $\overline{Z}_{21} = 0.803$ s; and (iii) the FN packet transfer times follow shifted gamma distributions with means $\overline{X}_{12} = 0.313$ and $\overline{X}_{21} = 0.145$ s. Note that according to the classification of the network delays, these values correspond to a condition of low network-delays. The free parameters of the system are the initial allocation and

the distribution of the failure times of the servers. The initial allocation was set to $m_1 = 50$ and $m_2 = 25$ tasks and failure times were assumed to follow exponential distributions with means 300 and 150 s.

The optimal DTR policy calculated using Algorithm 2 and a non-Markovian model for the two-server testbed is $l_{12}=26$ and $l_{21}=0$ tasks. Such policy provides a theoretical service reliability of 0.6007. Figure 4.3(c) shows theoretical predictions, MC simulations as well as experimental results for the service reliability of the two-server testbed. Results show the case when the optimal reallocation from server 2 to 1 is used ($l_{21}=0$), while different number of tasks are migrated from the first to the second server (l_{12}). In both simulations and experiments, the service reliability is calculated by averaging failure or success outcomes. A total of 10000 and 500 independent realizations of each policy have been employed in computing MC simulations and experimental results, respectively. Figure 4.3(c) shows a remarkable agreement between simulations and the non-Markovian theoretical predictions. Experimental results show also a fairly good agreement with the theoretical curves, where the relative error between predictions and experiments is less than 7%. Note that if no task reallocation is performed, the service reliability is reduced in approximately 15%. If a Markovian approximation is employed to devise the optimal DTR policy for the two-server testbed, the service reliability is reduced in approximately 1.5%.

In addition, the service reliability of the two-server DCS has also been optimized. First, it has been assumed that the DTR policy is executed at $t_b = 0$; therefore, Theorem 7 only must be used to solve the optimization problem (4.5). Figures 4.4(a) and (b) show the service reliability under for $l_{21} = 5$ and $l_{21} = 18$ are plotted as a function of l_{12}. On one hand, small values for l_{12} imply that server 1 remains unbalanced with respect to server 2 and executes most of its workload. As a consequence, the second server is under utilized because, on average, server 2 executes its entire workload before it fails. Therefore, the time required to serve the workload becomes

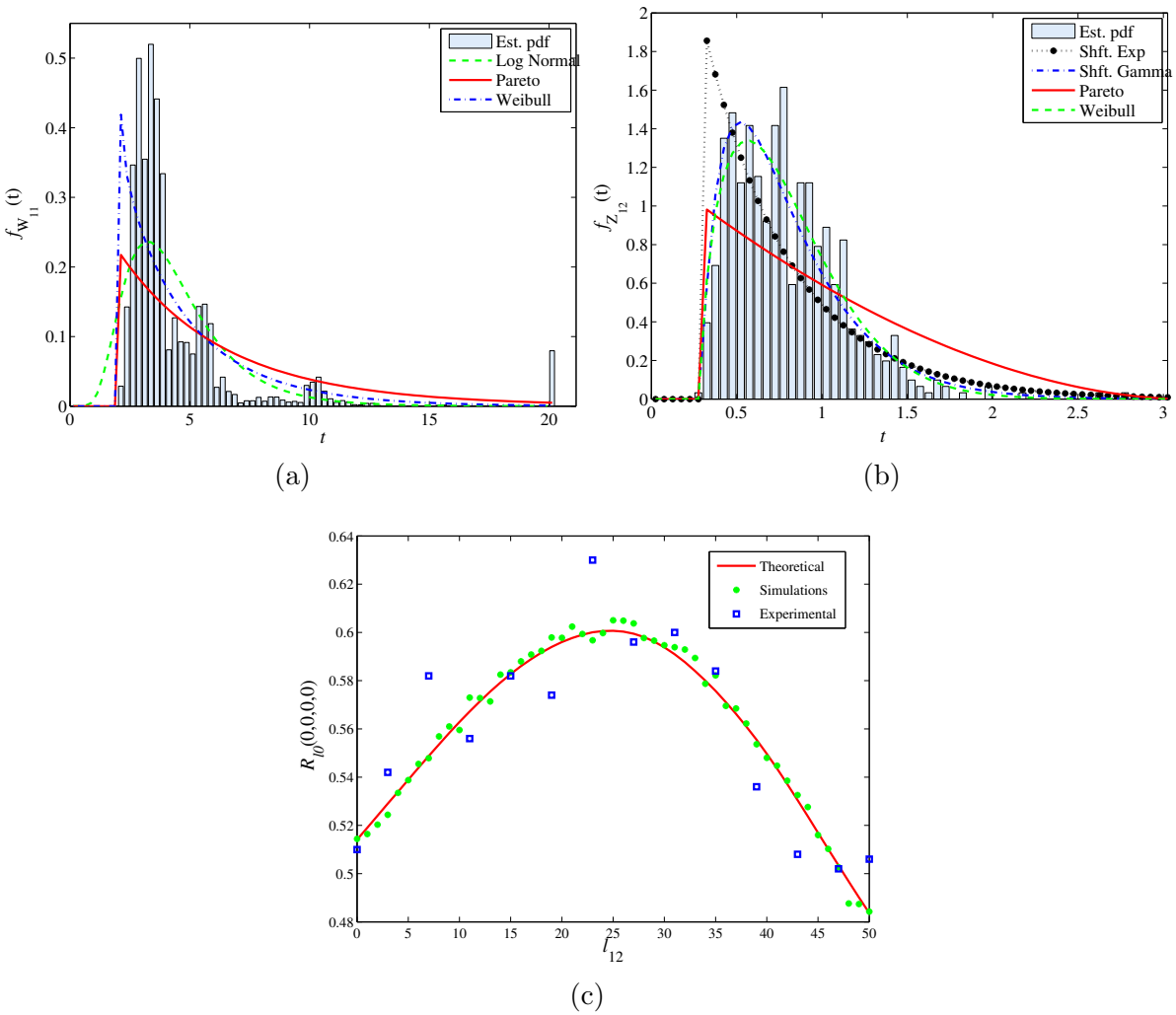


Figure 4.3: Normalized histogram and fitted pdf of: (a) Service time at server 1; and (b) Task transfer time from server 1 to 2. (c) Service reliability as a function of DTR policies.

"large" and the service reliability is "small." On the other hand, when l_{12} approaches to 60, the first server transfers approximately half of its initial load to the second server. Hence, almost all the tasks are queued and served at the less reliable server until it fails, thereby reducing the service reliability. In addition to the theoretical predictions, Fig. 4.4 shows the Markovian approximation, MC simulations as well as

experimental results. In simulations, the service reliability is calculated by averaging outcomes (failures or successes) from independent realizations of the policies. The values of reliability obtained via MC simulation plotted in Fig. 4.4 correspond to centers of 95% confidence intervals, for which the estimated service reliability will not differ from the true value by more than 0.0025. Simulation results strongly agree with the age-dependent theoretical predictions, and remarkably, experiments conducted on the two-server DCS show a fairly good agreement with theoretical curves. In the experiments, the service reliability is calculated by averaging the results of 500 independent trials for each policy shown in Fig. 4.4(a). It must be noticed that, in this case, the Markovian approximation yields fairly accurate predictions, where the relative approximation error is below 4%.

Next, the optimization problem (4.5) is solved to find the optimal time to trigger the DTR policy; Theorems 5 and 7 must be used in this case. Figure 4.3(b) shows theoretical predictions, MC simulations and experimental results for the service reliability as a function of the reallocation instant, for some representative reallocations. After solving (4.5), a maximal service reliability of 0.874 is achieved by the system at $t_b^* = 0$ by the following four DTR policies: $\mathbf{L}_1^* = \left(\begin{smallmatrix} 0 & 22 \\ 0 & 0 \end{smallmatrix}\right)$, $\mathbf{L}_2^* = \left(\begin{smallmatrix} 0 & 22 \\ 1 & 0 \end{smallmatrix}\right)$, $\mathbf{L}_3^* = \left(\begin{smallmatrix} 0 & 23 \\ 0 & 0 \end{smallmatrix}\right)$, and $\mathbf{L}_4^* = \left(\begin{smallmatrix} 0 & 23 \\ 1 & 0 \end{smallmatrix}\right)$. Figure 4.3(b) shows the service reliability as a function of t_b for the optimal policy $\mathbf{L}_1^*$. Note that an improper selection of the number of tasks to reallocate can produce a notorious reduction on the service reliability, as is depicted for the case of choosing $\mathbf{L} = \left(\begin{smallmatrix} 0 & 1 \\ 19 & 0 \end{smallmatrix}\right)$. Note also that, an improper selection of the number of tasks to reallocate can be compensated by delaying the DTR action.

Finally, the service reliability of a heterogeneous five-server DCS executing an application composed of $M = 150$ tasks has been maximized. The failure times of the servers have been assumed exponential with mean failure times of 350 s, 10 s, 20 s, 200 s, and 300 s, respectively. The service times were experimentally characterized in the testbed DCS, and as in the two-server case, service times follow

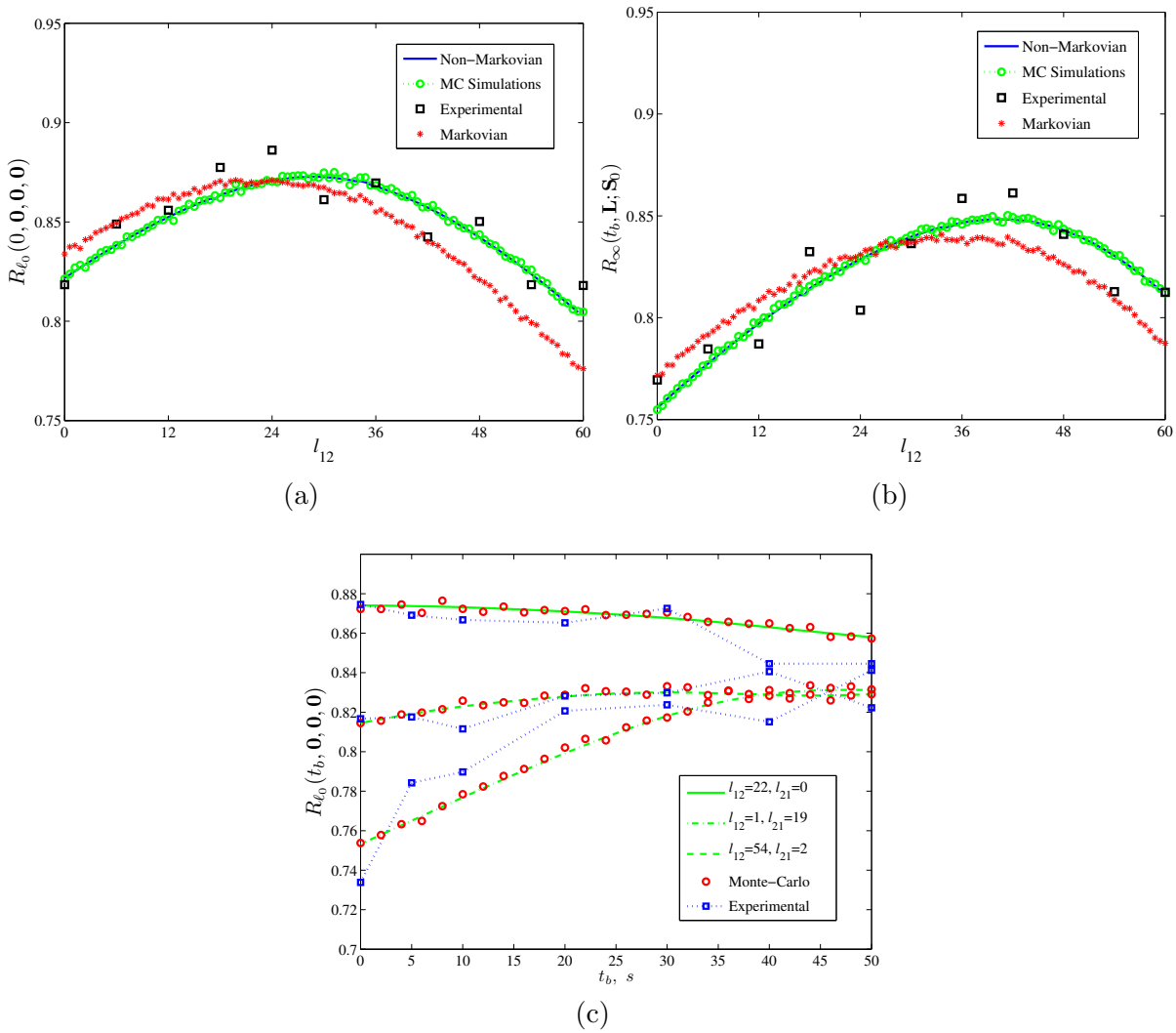


Figure 4.4: Service reliability as a function of the number of tasks exchanged from server 1 to 2, 1 when DTR policy is executed at $t_b = 0$. (a) $l_{21} = 5$ and (b) $l_{21} = 18$ tasks. (c) Service reliability as a function of the reallocation instant for four representative reallocations, $\mathbf{L}$.

Pareto distributions and transfer times follow shifted Gamma distributions. The channel-dependent parameters, also estimated from data collected using the testbed DCS, are listed in Table 4.3. The mean service times of the servers are 5.945, 2.009, 3.866, 3.943, and 5.447 s. For brevity, only the minimum and the maximum values

of the estimated mean arrival times of FN packets are provided: 0.343 and 7.727 s, respectively.

Table 4.3: Empirically characterized parameters, a_{jk} and b_{jk}, of the first-order approximation for the average task-transfer times in a five-server DCS.

a_{jk}	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
$j=1$	—	0.898	0.838	0.706	0.751
$j=2$	0.336	—	0.335	0.273	0.350
$j=3$	0.541	0.665	—	0.677	0.617
$j=4$	0.248	0.532	0.408	—	0.273
$j=5$	0.219	0.355	0.298	0.234	—
b_{jk}	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
$j=1$	—	1.970	2.219	2.000	2.199
$j=2$	1.651	—	1.993	1.876	1.667
$j=3$	5.001	4.997	—	5.203	5.557
$j=4$	4.131	7.604	5.862	—	7.604
$j=5$	3.009	2.887	2.731	2.943	—

For this DCS, three different DTR policies have been devised and discussed: (i) The *Null DTR policy* where tasks are not exchanged in the system; (ii) The *Full DTR policy* where tasks exchanged are given by the initial unadjusted values (4.15); and (iii) The *Reliability policy* where the number of tasks reallocated is calculated using Algorithm 2 with Λ_j parameters defined to be equal to the failure rate of the servers. Theoretical predictions obtained for different initial task allocations are listed in Table 4.4. The service reliability was calculated via simulations and values listed in Table 4.4 correspond to centers of 95% confidence intervals, for which the estimated reliability will not differ from the true value by more than 0.005. Also, the column labeled as "Experimental" presents results obtained after averaging 500 realization of experiments conducted on the testbed DCS. The first three rows of Table 4.4 represent cases where the system is totally imbalanced. The last four rows represent cases where the initial allocation was uniformly, according to servers' reliability, according to servers' processing speed, and arbitrarily.

It can be observed from Table 4.4 that the Reliability DTR policy outperforms

the other two policies in all the cases considered. It can also be noticed that the Reliability DTR policy effectively increases the service reliability because it trades off network queuing times and servers idle times by computing appropriate amount of tasks to exchange among the servers. For instance, when all the tasks are queued at the fourth server if no DTR action is executed, then on an average after 200 s the fourth server fails, while on an average the following events have occurred in the DCS: (i) the second and third servers have failed; (ii) the fourth server has been informed about the failures of the second and third servers; and (iii) the fourth server has executed 50 tasks. So, it can be clearly noticed that the first and the fifth server remained idle for long periods of time, and even worst, the second and third servers were never employed to serve any task. On the contrary, when the Full DTR policy is used, the fourth server decides to transfer 59, 1, 3, and 50 tasks to the first, second, third, and fifth server, respectively, while 37 tasks remain queued at the fourth server. From the previous discussion it becomes evident that Full DTR policy is advantageous over the Null policy, as evidenced by the service reliability shown in Table 4.4. Notably, the Reliability policy takes an even better by reallocating 38, 1, 3, and 41 tasks to the first, second, third, and fifth server, respectively. Note that by sending fewer tasks to the first and fifth server, the Reliability policy reduces the idle time of these servers as compared to the Full policy.

Table 4.4: Service reliability under different DTR policies.

Initial load $(m_1, \ldots, m_5)$	Service Reliability			
	Null	Full	Reliability	
	Theoretical	Theoretical	Theoretical	Experimental
(150,0,0,0,0)	0.210	0.508	0.509	0.527
(0,0,0,150,0)	0.330	0.532	0.583	0.575
(0,0,0,0,150)	0.255	0.533	0.543	0.559
(30,30,30,30,30)	0.634	0.557	0.634	0.603
(59,2,4,34,51)	0.534	0.555	0.556	0.539
(18,55,29,27,21)	0.642	0.563	0.642	0.625
(26,30,28,38,28)	0.642	0.563	0.642	0.603
(40,15,40,35,20)	0.621	0.562	0.624	0.649

The effect of the selection of various reallocation criteria on the service reliability is now studied. Three DTR policies have been considered, each one of them having a different reallocation criterion but sharing the same algorithm to compute the amount of tasks to reallocate. The Reliability LB policy reallocate tasks in the DCS according to the reliability of the servers. The *Processing-Speedpolicy* reallocate tasks in the DCS based upon the processing rate of the servers, i.e., $\Lambda_j = \lambda_{d_j}$. Finally, the *Maximal-Service policy* uses a balancing criterion that combines both processing and failure rates, that is, the Λ_j are defined as in (4.14). Additionally, a MC–based exhaustive search has been conducted, over the number of tasks to reallocate, in order to estimate the optimal service reliability for each case considered. The results of our evaluations are listed in Table 4.5.

Note that the fastest servers in the example are also the less reliable ones. Consequently, the task reallocation criterion employed by the Processing-Speed LB policy appears to be inappropriate in order to maximize the service reliability. However, it can be seen from Table 4.5 that, in most of the cases, the three policies achieve approximately the same performance, which shows the strength of our approach. For example, in the case when all the tasks are initially queued at the fourth server, the Processing-Speed policy dictates that 54 tasks have to be transferred to the second server. However, when Algorithm 2 is employed to calculate the number of tasks to reallocate, such amount reduces to only 11 tasks. From Table 4.5 it is observed that the Maximal-Service policy outperforms in almost all the cases the other two policies. This is because such policy trades off reliability and computing speed in both the reallocation criterion and the excess workload partitioning. Finally, it can be seen from Table 4.5 that the service reliability achieved by the policies is within 70% of the optimal service reliability for each case. In fact, the optimum is achieved in some cases.

Table 4.5: Service reliability achieved by three DTR policies, which have different task reallocation criteria. For comparison purposes, the optimal value obtained for each case is listed.

Initial load	Service reliability			
$(m_1, \ldots, m_5)$	Reliability	Processing-Speed	Maximal-Service	Optimum
(150,0,0,0,0)	0.509	0.511	0.573	0.631
(0,0,0,150,0)	0.583	0.533	0.612	0.615
(0,0,0,0,150)	0.543	0.566	0.613	0.619
(30,30,30,30,30)	0.634	0.603	0.636	0.657
(59,2,4,34,51)	0.556	0.608	0.638	0.668
(18,55,29,27,21)	0.642	0.623	0.640	0.649
(26,30,28,38,28)	0.642	0.639	0.642	0.642
(40,15,40,35,20)	0.624	0.610	0.643	0.656

4.3 Conclusions

An analytical, probabilistic framework to devise decentralized DTR polices that minimize the average execution time and maximize the QoS and the service reliability of heterogeneous DCSs has been presented in this chapter. The framework for devising DTR policies relies on the general analytical stochastic model developed in the previous chapter. Combining such a general model for DCSs with the general class of DTR policies employed here has provided three key insights about the behavior of heterogeneous DCSs. First, when incorrect models for the random times governing the dynamics of the system are employed to calculate DTR policies, not only inaccurate predictions are yielded for the performance and reliability metrics, but also the calculated DTR policies specify inappropriate amount of tasks to reallocate among the servers, which, in turn, reduce the metrics under study. Second, there is a fundamental trade off between minimizing the average service time and maximizing service reliability, such a trade off can be obtained, for instance, by devising DTR policies that optimize simultaneously the two metrics. Third, effective DTR policies for improving performance and reliability of DCSs must consider the heterogeneous processing capabilities of the servers, their reliability and the heterogeneous network

transfer delays.

When DTR is performed as dictated by the policies presented here, the service reliability can be improved up to 63% as compared to the reliability provided by a DCS, and up to 45% as compared to policies that consider server' reliability but disregard the communication costs over the network. Moreover, the suboptimal algorithm developed to compute the DTR policies achieves a service reliability within 80% of the optimal service reliability, and in cases achieves the optimal value. Using the framework presented here, the service reliability of a small-scale testbed DCS has been predicted and DTR policies to enhance its reliability have been devised. Such policies were implemented in the testbed and experimental results were compared to theoretical predictions. Evaluations have shown not only an improvement in the service reliability of the testbed, but also the remarkable accuracy of the predictions of the non-Markovian model developed in this dissertation. In terms of scalability, the algorithm for finding the number of tasks to reallocate among the servers scales linearly with the number of servers of the DCS and exploits the exact analytical age-dependent model for two-server DCSs.

Chapter 5

Modeling spatially correlated failures

This chapter presents a stochastic model for spatially correlated failures on DCSs. The model captures the logical and geographical spatial interactions of the servers in a DCS. Unlike the model for DCSs presented in the previous chapters, here the underlying network topology of a DCS is modeled by using graph theory. A MRFs theory is exploited in conjunction with the topological structure of the network to introduce "local specifications" of failures. These local specifications, in turn, induce a global distribution function of failure patterns to all the servers in the DCS. For completeness, the beginning of this chapter provides the required notation and terminology related to graph theory and MRFs.

5.1 Preliminaries

5.1.1 Graph theory and the network topology of a DCS

Physical as well as logical network topologies are commonly abstracted in the literature by means of graph theory. Server nodes are represented by a set of vertices and the (physical and/or logical) relationships between the servers are represented by edges linking pairs of them. Here, the underlying topology of an n-server DCS is described by the connected, undirected graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$ is the set of server nodes and $E \subset V \times V$ is the set of communication links in the network. Since the set of edges describes only if a connection exists or not between any pair of servers, a set of non-negative weights, $W(E)$, associated with E has been also defined. Specifically, the map $W : E \to [0, \infty)$ associates to each edge e in the graph a positive weight denoted as $W(e)$. This weight function allows to the model more general relationships between the servers, such as, geographical distances between them and communication costs among others.

For $v \neq u$, a trajectory from the server v to the server u on the graph $G = (V, E)$ is the non-empty, acyclic subgraph of G denoted by $T_{v,u} = (X, Y)$ that links servers $v = v_0$ and $u = v_k$. Formally, a trajectory is defined as $T_{v,u} = (X, Y) \subset G$ where $X = \{v_0, v_1, \ldots, v_k\}$ is the set of all the servers visited in the trajectory from node v_0 to server v_k, while $Y = \left\{ \{v_0, v_1\}, \{v_1, v_2\}, \ldots, \{v_{k-1}, v_k\} \right\}$ is the set with the sequence of edges employed in the trajectory. For convenience, it has been defined that $T_{v,v} = \emptyset$. Let $\mathcal{T}_{v,u}$ denote the collection of all possible trajectories between servers v and u. With this, the logical distance, $d_L(v, u)$, as well as the weighted

distance, $d_W(v,u)$, between servers v and u in G can be defined as:

$$d_L(v,u) \triangleq \min_{(X,Y)\in\mathcal{T}_{v,u}} |Y|, \tag{5.1}$$

$$d_W(v,u) \triangleq \min_{(X,Y)\in\mathcal{T}_{v,u}} \sum_{e\in Y} w(e), \tag{5.2}$$

where $|Y|$ is the cardinality of the set Y.

5.1.2 Neighborhood of a server

Servers can be related to some group of servers via the concept of "neighborhood." A neighborhood is a relationship between servers satisfying the following properties:

(i) A server is not a neighbor of itself: for all $v \in V,\ v \notin \mathcal{N}_v$,

(ii) A neighboring relation is reflexive: for all $v, u \in V,\ v \in \mathcal{N}_u \Leftrightarrow u \in \mathcal{N}_v$,

where $\mathcal{N}_v$ denotes the neighborhood of server v. For instance, the neighborhood of the server v can be defined as:

$$\mathcal{N}_v \triangleq \{u : d_W(v,u) \leq D_{\max}\ u, v \in V\}, \tag{5.3}$$

where the positive number $D_{\max}$ is a parameter that can extend or reduce the size of a neighborhood. Note that if $D_{\max} = 1$ and $d_W(v,u)$ is replaced by $d_L(v,u)$ in (5.6), then the definition of neighborhood reduces to the traditional definition of nearest neighbors on a graph. For notational convenience, the set $\widetilde{\mathcal{N}}_v \triangleq \mathcal{N}_v \cup \{v\}$ is also introduced. Finally, the neighborhood system, $\mathcal{N}$, induced by the graph G is defined as the collection of all neighborhoods, i.e., $\mathcal{N} = \{\mathcal{N}_v,\ v \in V\}$.

5.1.3 Random fields and Markov random fields

Definition. Let X_i denote a discrete random variable defined on $\Omega_i = \{0, 1, \ldots, N_i\}$. For $V = \{1, 2, \ldots, n\}$, let $\mathbf{X} = \{X_i,\ i \in V\}$ be a the collection of random variables

taking values on $\Omega = \Omega_1 \times \ldots \times \Omega_n$, termed as the "configuration space." A probability measure π is a *random field* if for all ω in Ω the probability $\pi(\omega)$ is positive, [13,53,80].

Intuitively, a random field can be regarded as a "random variable" taking values on the configuration space, where a particular value of a configuration will be denoted as $\mathbf{x} = \big(x_1\, x_2\, \ldots\, x_n\big)$, with $x_i \in \Omega_i$ for all $i \in V$. Further, the restriction of $\mathbf{x}$ to the set A, for $A \subset V$, is defined as $\mathbf{x}(A) = \{x_i,\, i \in A\}$.

Definition. *Gibbs fields* is a random field that follows the Gibbs distribution:

$$\pi_{\mathbf{X}}(\mathbf{x}) = \frac{1}{Z_T} \exp\left(-\frac{\mathcal{E}(\mathbf{x})}{T}\right), \tag{5.4}$$

where Z_T a normalizing constant, termed as the partition function, T is a constant termed as the temperature and $\mathcal{E}(\mathbf{x})$ is referred to as the energy function, [13,53,80].

Definition. Let $\mathbf{X}$ be a random field. This random field is a *Markov random field* with respect to a neighborhood system $\mathcal{N}$ induced by the graph $G = (V, E)$ if for all $v \in V$ the random variables X_v and $\mathbf{X}(V \setminus \widetilde{\mathcal{N}}_v)$ are independent given $\mathbf{X}(\mathcal{N}_v)$. More precisely, for all $v \in V$ and $\mathbf{x} \in \Omega$

$$\mathsf{P}\{X_v = x_v | \mathbf{X}(V \setminus v) = \mathbf{x}(V \setminus v)\} = \mathsf{P}\{X_v = x_v | \mathbf{X}(\mathcal{N}_v) = \mathbf{x}(\mathcal{N}_v)\}. \tag{5.5}$$

According to the Hammersley-Clifford theorem, there is an equivalence between MRFs and certain types of Gibbs' random fields [13,53,59]. When the energy function can be written as $\mathcal{E}(\mathbf{x}) = \sum_{C \in \mathcal{C}} \mathsf{V}_C(\mathbf{x})$, where $\mathsf{V}_C(\mathbf{x})$ is a Gibbs potential on the clique[1] C, then a MRF is equivalent to a Gibbs field [13, 53, 59]. (The variable $\mathcal{C}$ denotes the collection of all cliques.)

[1]Formally, a subset C of V with one or more elements is called a clique of the graph $G = (V, E)$ if and only if any two distinct members of C are mutual neighbors.

5.2 Model for spatially correlated failures

The key idea to develop a model for spatially correlated failures is as follows. First, to capture the logical and geographical connections between the servers in a DCS, the underlying topology of the communication network connecting the servers has been modeled using graph theory. Next, the ability of MRFs to model spatially correlated phenomena has been exploited by defining meaningful local interactions that are simple to specify. These interactions in turn, define a global Gibbs distribution of spatially correlated failures. The technical details of the model are provided next.

5.2.1 Markov random fields approach for modeling spatially correlated failures

Suppose that the undirected graph $G = (V, E)$ represents the topology of a DCS, where $V = \{1, \ldots, n\}$ is the set of servers and $E \subset V \times V$ represents the underlying topology of the communication network connecting the servers. In order to capture both geographical as well as logical correlations in a MRF setting, the following neighborhood system is defined:

$$\mathcal{N}_v \triangleq \{u : d_W(v, u) \leq D_{\max} \text{ or } d_L(v, u) = 1,\ u, v \in V\}, \tag{5.6}$$

In words, two servers are neighbors if their Euclidean (geographical) distance is within the range $D_{\max}$ or if they have a direct connection with each other. From this definition of neighborhood, the graph G induces the neighborhood system $\mathcal{N}$.

Suppose now that X_i is a binary random variable representing if a server has failed ("1") or not ("0"). The definition of neighborhood-system in conjunction with the collection of binary random variables $\mathbf{X} = \{X_i,\ i \in V\}$ taking values on the configuration space $\Omega = \{0, 1\}^n$ is employed here to introduce a MRF. The definition of the MRF is complete when the Markovian condition (5.5) is specified. That is, the

MRF is completely determined when the likelihood of failure of a server, conditional on the failed or working state of its neighbor servers is specified.

Requirements. It is of interest to this dissertation analyzing the performance of DCSs in harsh scenarios where the failure of a server induces failures in other functioning servers, for instance, due to the inability of the working servers to exchange data and information with a failed server. It is also of interest to this work to model situations where the geographical or logical proximity of a functioning server to a failed server increases the likelihood of failure on the functioning server and its neighbor nodes. To fulfill all these requirements, the following local specification for the likelihood of failure of server v given the failed or working state of its neighbor servers is proposed

$$\mathsf{P}\{X_v = x_v | \mathbf{X}(\mathcal{N}_v) = \mathbf{x}(\mathcal{N}_v)\} = \frac{\exp\left(-T^{-1}\, x_v (r_v - \sum_{u \in \mathcal{N}_v} s_{v,u}\; x_u)\right)}{1 + \exp\left(-T^{-1}(r_v - \sum_{u \in \mathcal{N}_v} s_{v,u}\; x_u)\right)}, \tag{5.7}$$

where T is a constant, r_v is a non-negative parameter modeling the resilience of the vth server to failures and $s_{v,u}$ is a non-negative parameter modeling the strength of interaction between the servers u and v.

Note that, due to the summation in (5.7), the likelihood of server v of being in a failed state, $x_v = 1$, effectively increases when one or more of its neighboring servers are also in a failed state, $x_u = 1$. Moreover, consider the following definition for the strength of interaction parameters:

$$s_{v,u} = \begin{cases} \frac{D_{\max}}{d_W(v,u)} + s_L & , \text{ if } u \in \mathcal{N}_v \text{ and } d_W(v,u) \leq D_{\max} \\ s_L & , \text{ if } u \in \mathcal{N}_v \text{ and } d_W(v,u) > D_{\max} \\ 0 & , \text{ if } u \notin \mathcal{N}_v \end{cases}, \tag{5.8}$$

where s_L is a non-negative parameter modeling the logical strength of interaction between nodes v and u. This inhomogeneous definition for $s_{v,u}$ clearly increases the likelihood of failure of servers when they are geographically or logically close to failed servers.

The equivalence between MRFs and Gibbs fields can be exploited to determine the energy function. By invoking the law of total probability, the definition (5.7) and recalling the Markovian condition (5.5) it is straightforward to obtain:

$$\mathcal{E}(\mathbf{x}) = \sum_{v \in V} r_v \; x_v - \sum_{v \in V} \sum_{u \in \mathcal{N}_v} s_{v,u} \; x_v x_u, \tag{5.9}$$

which is written in terms of second-order Gibbs potentials. The energy function can be written using matrix-vector notation as: $\mathcal{E}(\mathbf{x}) = \mathbf{x}^T\mathbf{r} - \mathbf{x}^T\mathbf{A}\mathbf{x}$, where $\mathbf{x} = (x_1 \dots x_n)^T$, $\mathbf{r} = (r_1 \dots r_n)^T$, and $\mathbf{A} = (s_{v,u})_{n \times n}$. Thus, the Gibbs distribution associated with this energy function is

$$\begin{aligned} \pi_{\mathbf{X}}(\mathbf{x}) &= \frac{1}{Z_T} \exp\left(- \frac{\sum_{v \in V} r_v \; x(v) - \sum_{v \in V} \sum_{u \in \mathcal{N}_v} s_{v,u} \; x_v x_u}{T} \right) \\ &= \frac{1}{Z_T} \exp\left(- \frac{\mathbf{x}^T\mathbf{r} - \mathbf{x}^T\mathbf{A}\mathbf{x}}{T} \right). \end{aligned} \tag{5.10}$$

Note that the local specification (5.7) is independent of the normalizing constant Z_T, while the Gibbs distribution depends on it. Note also that when the strength of interaction parameters are equal to zero, the Gibbs distribution reduces to the case of independent failures.

5.3 Monte-Carlo approach for sampling spatially correlated failures

5.3.1 The Gibbs sampler

Realizations of spatially correlated failures following a Gibbs distribution can be sampled, in theory, from (5.10). Unfortunately, the normalizing constant T is usually hard to compute since due to the large dimension of the configuration space. In order to circumvent this problem, sampling algorithms such as Gibbs or Metropolis samplers can be employed to generate realizations of (5.10). These sampling algorithms

yield realizations of MRFs by constructing a field-valued, homogeneous Markov chain that has as its stationary distribution, the desired Gibbs distribution. The idea of the algorithm is to generate a realization of a Markov chain that, at a large number of iterations, will be close to (5.10). A key result in MRFs theory proves that a Markov chain having as a stationary distribution (5.10) can be constructed using the local specifications, [13, 59].

From (5.7), the local specifications for the vth server are:

$$p_0 = \pi\left(0|\mathbf{x}(\mathcal{N}_v)\right) = \frac{1}{1+\exp\left(-T^{-1}(r_v - \sum_{u\in\mathcal{N}_v} s_{v,u}\ x_u)\right)}, \tag{5.11}$$

$$p_1 = \pi\left(1|\mathbf{x}(\mathcal{N}_v)\right) = \frac{\exp\left(-T^{-1}\left(r_v - \sum_{u\in\mathcal{N}_v} s_{v,u}\ x_u\right)\right)}{1+\exp\left(-T^{-1}(r_v - \sum_{u\in\mathcal{N}_v} s_{v,u}\ x_u)\right)} = 1 - p_0. \tag{5.12}$$

Once these expressions are known, the Gibbs sampler can be implemented. Algorithm 3 shows the details of the sampling process, whose main idea is the following: Starting with an initial random configuration, at each iteration of the algorithm a server is randomly picked, say the vth server. The value of the realization x_v associated with the random variable X_v, is updated according to either p_0 or p_1. This process is repeated a large number of times, K, and as a result of these K iterations a sample from (5.10) is obtained.

5.3.2 Sampled patterns of spatially correlated failures

To demonstrate the ability of the MRF-based model for generating correlated failures, DCSs with representative network topologies have been considered. In the examples, a nationwide DCS has been considered where servers are located at several cities in the US as shown in Fig. 5.1. The network topology of the first DCS considered is a realization from the class of the so-called random networks. In the second and third DCSs considered, the underlying communication networks correspond to modified versions of the AT&T IP backbone network 2Q2000, [31]. The DCSs studied in

Algorithm 3 Gibbs sampler for the distribution (5.10)

Require: $G = (V, E)$, T, r_v, s_L, $D_{\max}$, $\mathcal{N}_v$, and K

Ensure: $\mathbf{x}$

```
Set x^0 to any random value in Λ^V
Set k = 0
while k ≤ K do
    x^{k+1} ← x^k
    Randomly pick v ∈ V
    Compute p_0 using (5.11)
    Generate a random number α ~ U[0, 1]
    if α < p_0 then
        Set x_v^{k+1} = 0
    else
        Set x_v^{k+1} = 1
    end if
    k ← k + 1
end while
x ← x^k
```

this section comprise 20, 38 and 17 servers, and the Fiedler connectivity[2] of the communication networks are 0.47, 0.45 and 0.23, respectively.

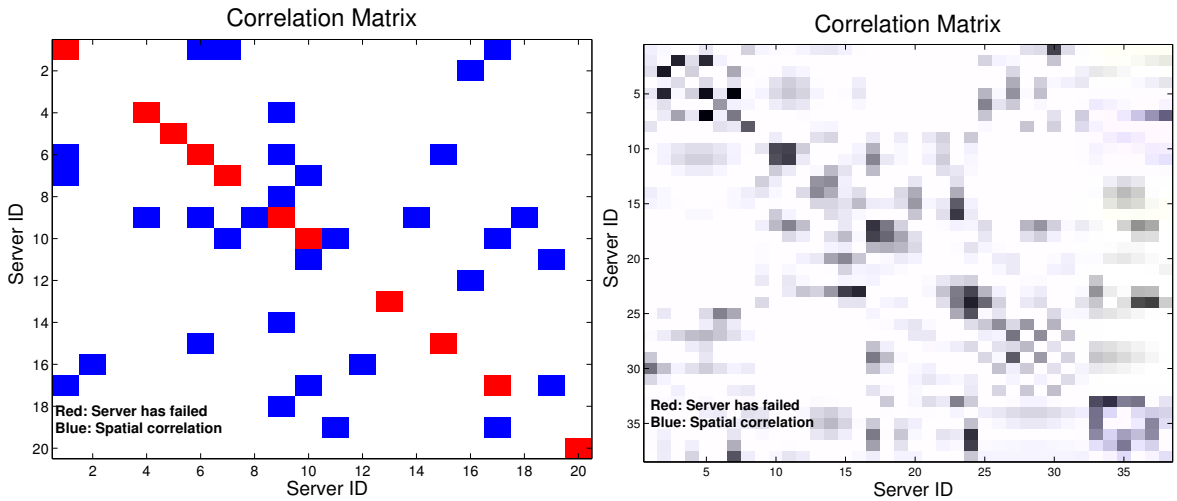
Unless otherwise is stated, the following parameters have been used to generate spatially correlated failures on the DCS: $T = 1$, $r_v = 2$, $s_L = 1$, and $D_{\max} = 300$ miles. Additional parameters employed are: (i) the Gibbs sampler iterates $K = 50000$ times before yielding a sample of the MRF; and (ii) covariance matrices were estimated using 2000 realizations of the MRF. It has been termed here as the (spatially) independent distribution to a Gibbs distribution where the

[2]The Fiedler connectivity is defined as the second smallest eigenvalue associated with the Laplacian matrix of the graph G modeling the topology of the network.

Figure 5.1: (a) Sample DCS composed of 20 servers. (b) DCS interconnected by means of the AT&T IP backbone network 2Q2000, [31]. (c) DCS interconnected by means of a simplified version of the AT&T IP backbone network 2Q2000, [31].

only cliques considered are those having a single node, that is, when $s_{v,u} = 0$ for all v and u in (5.10).

Spatial correlation in the failure patterns has been tested by generating a total of 2000 failure realizations, and sampled covariance matrices have been computed. Each off-diagonal element of such matrices was statistically tested for correlation using a t-test for the hypothesis of no correlation with a confidence of 99%. The results of these tests and a sample realization of the failures in the network were used to construct the correlation matrices shown in Fig. 5.2. The elements in the diagonal of



(a) (b)

Figure 5.2: Matrices showing the spatial correlation in the case of the (a) sample network with 20 nodes, and (b) the AT&T IP backbone network 2Q2000. The elements in the diagonal of the matrices correspond to a sample realization from the Gibbs distributions. A red color means a failed server. The off diagonal elements show if there is (blue color) or not (white color) spatial correlation between the failures at the servers.

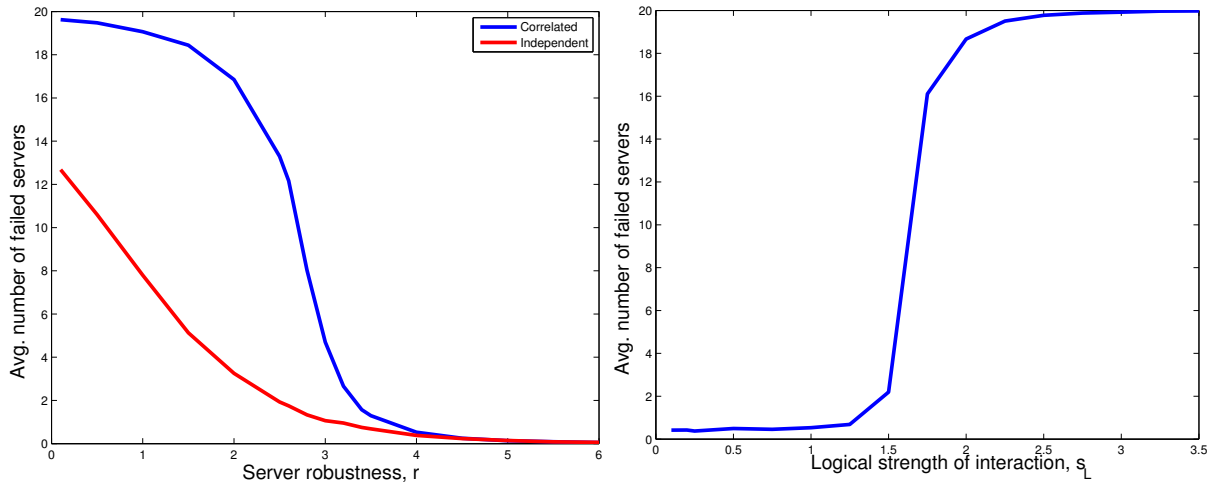
the matrices correspond to sample realizations from the Gibbs distributions obtained from the Gibbs sampler described in Algorithm 3. A server, say, the ith server has become failed if the ith diagonal element is "1" (red color) and is in a working state if the ith element is "0" (white color). The off-diagonal elements of the matrices shown in Fig. 5.2 represent if there is spatial correlation (value "1" in blue color) or not (value "0" in white) between the ith and jth servers.

Figures 5.3 and 5.4 show the average number of failed servers as a function of the server robustness, the logical strength of interaction between servers, and the maximum geographical distance between neighboring servers for the two DCSs depicted in Fig. 5.1(a) and (b). It can be observed from the figures that the average number of failed servers increases as the robustness parameter decreases. In addi-

tion, as the logical strength of interaction or the $D_{\max}$ parameters increase so it does the average number of failed servers. This behavior suggests that failures propagate more intensely as these two parameters increase. As expected, when the robustness parameter is fixed, the average number of failed servers is larger in the case of spatially correlated failures compared to the case of a spatially independent failures. Finally, note that the slopes of the plots in Figs. 5.3(a) to (c) are steeper than those shown in Figs. 5.4(a) to (c). This is attributed to the connectivity of the underlying networks associated with the DCSs. Since the 20-server DCS is more connected than the 38-server DCS, according to the Fiedler eigenvalue, it is expected that the spatial interaction between servers is naturally accentuated due to the larger number of relative connections in the 20-server DCS as compared to the connections in the 38-server DCS.

Table 5.1 compares the effect of correlation parameters s_L and $D_{\max}$ on some interesting failure patterns for the 17-server DCS shown in Fig. 5.1(c). The normalizing constant has been calculated by considering all the values in the configuration space for independent and correlated failures. With this, the probability of each specific failure pattern can be calculated from the Gibbs distribution (5.10). Results show that the probability of having a large fraction of the servers failing is much higher in the correlated-failure case than in the independent-failure case. As expected from such model, the correlation parameters $s_{v,u}$ can be used to control the degree of failed-servers clustering or bunching. Similarly, the probability of failure patterns with very few failed servers is much lower in the correlated-failure scenario than that corresponding to the independent case. Namely, there is a weaker "inhibition" effect in the correlated-failure scenarios compared to the independent-failure scenario.

Finally, Figure 5.5 shows the empirical distribution of failures on the servers for two different selections of the parameter r_v. For simplicity of visualization, Gaussian distribution functions have been placed at the failed servers with the height equal to



(a) (b)

(c)

Figure 5.3: Average number of failed servers versus (a) r_v parameter, (b) s_L parameter, and (c) $D_{\max}$ parameter for the DCS with 20 servers shown in Fig. 5.1(a).

the number of times a server has failed in the entire 2000 realizations. In the model used to generate the failures in Fig. 5.5(a), all servers have a common r_v value of 2.6 with the exception of server 2, which is set to $r_2 = 0.1$. Similarly, for Fig. 5.5(b), $r_8 = 0.1$ while all the other servers share the same value of 2.6 for r_v. Clearly, there is a higher probability of failure at server 2 and its neighbors in Fig. 5.5(a) than that

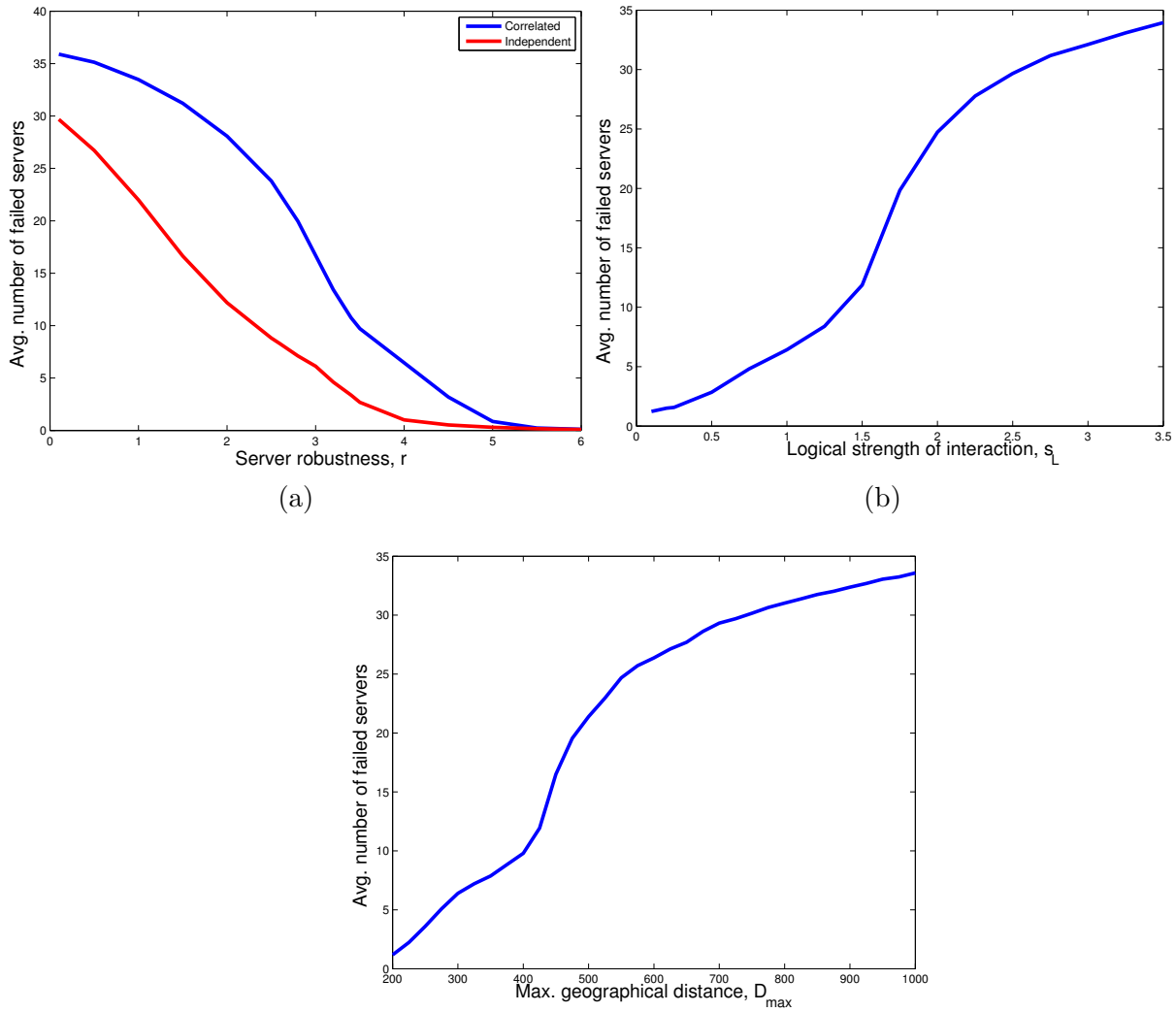
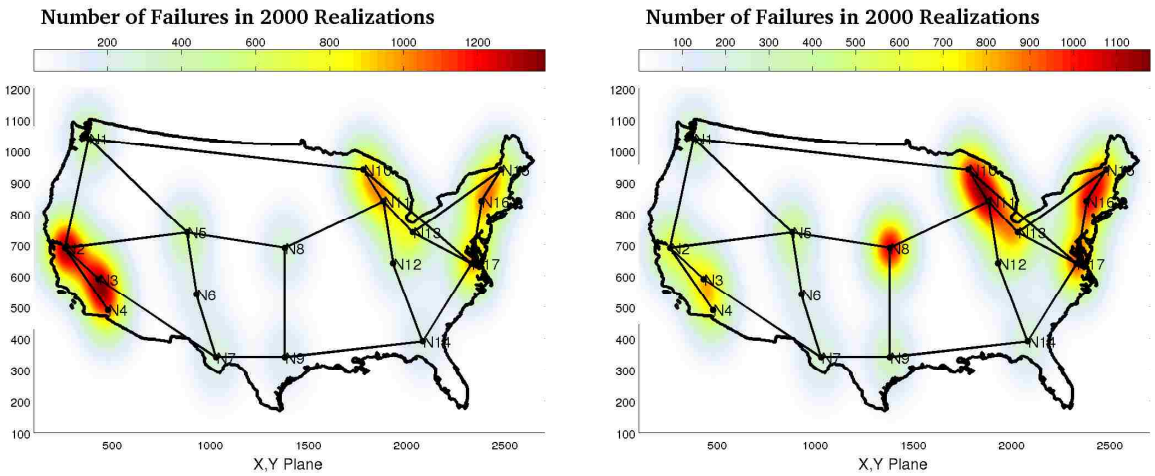


Figure 5.4: The average number of failed servers versus (a) r_v parameter, (b) s_L parameter, and (c) $D_{\max}$ parameter for the DCS with 38 servers shown in Fig. 5.1(b).

in Fig. 5.5(b). The same behavior is observed at server 8 for Fig. 5.5(b) compared to Fig. 5.5(a).

Table 5.1: Failure patterns in correlated and independent failure scenarios for the DCS shown in 5.1(c).

	Probability of failure patterns		
	Correlated	Independent	Failure pattern
	0.081	10^{-42}	All servers
Clustering	0.063	10^{-35}	All except 2, 4, 10, 11
	0.030	10^{-41}	All except server 17
Effect	0.030	10^{-41}	All except server 7
	0.030	10^{-41}	All except server 2
Inhibition	10^{-5}	0.006	One server only
	4×10^{-5} to 10^{-7}	4×10^{-5}	Two servers only
Effect	5×10^{-5} to 10^{-9}	4×10^{-7}	Three servers only



(a) (b)

Figure 5.5: Distribution of the failures on the DCS shown in 5.1(c). Failures are visualized with a Gaussian distribution function with height equal to the number of failures on each server of the topology. The distribution of the failures has been obtained by fixing $r_v = 2.6$ for all servers and changing only (a) $r_2 = 0.1$; and (b) $r_8 = 0.1$.

5.4 Conclusions

A novel framework, based on MRFs and graph theory, for modeling correlated failures of servers in DCSs have been developed. The model abstracts the arbitrary topology of the underlying network connecting the servers in a DCS. The developed failure model captures the spatial correlation between servers with logical and geographical connections, therefore capturing the percolation effect of node damage across the DCS. The model is developed by defining local conditional specifications of failure probabilities that are related directly to both the geographical and logical neighborhood relations imposed by the topology of the DCS. Key in the development of the model are the set of parameters termed as the strength of interaction between servers. These parameters specified the degree of interaction between servers in terms of physical distances and also in terms of logical coupling.

The statistical analysis conducted on realizations obtained from the model for correlated failures has shown that the failure of a single server does propagate to other functioning servers, depending on the intensity of the so-called inter-server strength of interaction parameter, which captures both logical and geographical relations between each pair of serves. The analysis confirms that the average number of failures increases as the logical and geographical strength of interaction between servers increases. As expected, the analysis confirms also that the average number of failed servers increases when correlated failures affect the servers of a DCS, as compared to the case of independent failures. Analytical results show also that the probability of having a failure pattern involving a large fraction of the servers is much higher when correlated failures affect the system, than in the case of independent failures. Moreover, the strength of interaction parameters specified in the model can be used to control the degree of failed-servers clustering or bunching. This result is of practical interest in order to identify the vulnerabilities associated with coordinated attacks on the network infrastructure of the DCS.

Chapter 6

Robust distributed computing in the presence of correlated failures

In this chapter the analytical age-dependent models for service reliability as well as the model for spatially correlated failures are combined to assess the reliability of DCSs in the presence of spatially correlated failures. The main assumption imposed is that servers affected by a pattern of spatially correlated failures will fail simultaneously, and a temporal distribution for the cluster of failed servers in order to exploit the age-dependent characterization of the service reliability. A DTR policy considering spatially correlated failures is also presented in this Chapter.

6.1 Characterizing reliability in the presence of correlated failures

In order to calculate the service reliability in the presence of spatially correlated failures, a hybrid analytical and Monte-Carlo approach is presented. By means

of a Gibbs sampler, a large number of realizations of spatially correlated failures can be generated. Thus, conditional on a particular failure pattern, the analytical model for the DCS can be employed to calculate the service reliability. Finally, the calculated values for reliability are averaged over the number of realizations to obtain an estimate of the service reliability of a DCS in the presence of spatially correlated failures. The hybrid approach for calculating the service reliability is completed once a distribution for the random time to fail for the conditional set of servers is defined.

Conditional on the occurrence of a spatially correlated failure, let the random variable Y_I, be the random failure time of all those servers, indexed by the set I, that fail in a correlated manner. The same principles presented in Section 3.2.1 can be applied to obtain regenerative age-dependent equations for the conditional service reliability of a DCS. To this end, consider the following the Assumption A5 and associate a single age-variable, a_{C_I}, to the random time Y_I

Assumption A5. Suppose that the pdf of the random time Y_I is known and denoted as $f_{Y_I}(t)$. Suppose also that this random time is mutually independent to all the random times listed in Assumptions A1.

Theorem 13 is presented without proof as it follows the same principles of the proof of Theorems 1 and 2.

Theorem 13. *Consider an n-server DCS whose servers perform a synchronous DTR action at the time $\xi \geq 0$. Suppose that I denotes a set of servers that fail altogether in a correlated manner, and let $\mathbf{X} = \mathbf{x}_I$ denote the pattern of failed servers. For any $\ell \in \mathcal{I}$, the service reliability in executing an application, when the n-servers in a DCS perform a synchronous DTR action at the time $\xi \geq 0$, satisfies the system of*

recursive, coupled integral equations in ξ:

$$R_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C|\mathbf{X}=\mathbf{x}_I) = \int_0^\xi \Bigg[\sum_{i\in\mathcal{V}} R_{\ell_i}\Big(\xi-\alpha, (\mathbf{a}_M+\alpha)^{(i)}, \mathbf{a}_F+\alpha, \mathbf{a}_C+\alpha|\mathbf{X}=\mathbf{x}_I\Big)$$
$$\times G_{W_{i1}}(\alpha) + G_{Y_I}(\alpha) R_{\ell'_I}\Big(\xi-\alpha, \mathbf{a}_M+\alpha, (\mathbf{a}_F+\alpha)^{(I)}, \mathbf{a}_C+\alpha|\mathbf{X}=\mathbf{x}_I\Big)$$
$$+ \sum_{i=1}^n \sum_{j=1, j\neq i}^n G_{X_{ij}}(\alpha) R_{\ell_{ij}}\Big(\xi-\alpha, \mathbf{a}_M+\alpha, \mathbf{a}_F+\alpha, \mathbf{a}_C+\alpha|\mathbf{X}=\mathbf{x}_I\Big)\Bigg] d\alpha$$
$$+ \big(1 - F_{\tau_{\mathbf{a}}|\mathbf{X}}(\xi|\mathbf{x}_I)\big) R_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C|\mathbf{X}=\mathbf{x}_I) \tag{6.1}$$

where recursions are carried out in the discrete variables $\mathbf{m}$ and $\mathbf{F}$, the vectors $\mathbf{m}$ and $\mathbf{a}_M$ and the matrices $\mathbf{F}$, $\mathbf{0}$, $\mathbf{a}_F$, and $\mathbf{a}_C$ denote an arbitrarily specified initial system configuration, $\ell = h(\mathbf{m}, \mathbf{F}, \mathbf{0})$, $\ell_i = h(\mathbf{m}-\boldsymbol{\delta}_i, \mathbf{F}, \mathbf{0})$, $\ell'_I = h(\mathbf{m}, \mathbf{F}^{(I)}, \mathbf{0})$, $\ell_{ij} = h(\mathbf{m}, \mathbf{F}^{(ji)}, \mathbf{0})$, the vector $\mathbf{v}^{(i)}$ (respectively, the matrix $\mathbf{A}^{(ij)}$) is identical to the vector $\mathbf{v}$ (respectively, the matrix $\mathbf{A}$) but with its ith (respectively, ijth) component set to zero, and $R_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C|\mathbf{X}=\mathbf{x}_I)$ is the initial condition related to the ℓth integral equation. These initial conditions satisfy the system of recursive, coupled integral equations:

$$R_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C|\mathbf{X}=\mathbf{x}_I) = \int_0^\infty \Bigg[\sum_{i=1}^n R_{\ell_i}\Big(0, (\mathbf{a}_M+\alpha)^{(i)}, \mathbf{a}_F+\alpha, \mathbf{a}_C+\alpha|\mathbf{X}=\mathbf{x}_I\Big)$$
$$\times G_{W_{i1}}(\alpha) + G_{Y_I}(\alpha) R_{\ell'_I}\Big(0, \mathbf{a}_M+\alpha, (\mathbf{a}_F+\alpha)^{(I)}, \mathbf{a}_{C_0}+\alpha|\mathbf{X}=\mathbf{x}_I\Big)$$
$$+ \sum_{i=1}^n \sum_{j=1, j\neq i}^n G_{Z_{ji}}(\alpha)\, R_{\ell_{ji}}\Big(0, \mathbf{a}_M+\alpha, \mathbf{a}_F+\alpha, \mathbf{a}_C+\alpha|\mathbf{X}=\mathbf{x}_I\Big)$$
$$+ \sum_{i=1}^n \sum_{j=1, j\neq i}^n G_{X_{ij}}(\alpha) R_{\ell_{ij}}\Big(0, \mathbf{a}_M+\alpha, \mathbf{a}_F+\alpha, \mathbf{a}_C+\alpha|\mathbf{X}=\mathbf{x}_I\Big)\Bigg] d\alpha, \tag{6.2}$$

where recursions are carried out in the discrete variables $\mathbf{m}$, $\mathbf{F}$, and $\mathbf{C}$, the vectors $\mathbf{m}$ and $\mathbf{a}_M$ and the matrices $\mathbf{F}$, $\mathbf{0}$, $\mathbf{a}_F$, and $\mathbf{a}_C$ denote an arbitrarily specified initial system configuration, $\ell = h(\mathbf{m}, \mathbf{F}, \mathbf{C})$, $\ell_i = h(\mathbf{m}-\boldsymbol{\delta}_i, \mathbf{F}, \mathbf{C})$, $\ell'_I = h(\mathbf{m}, \mathbf{F}^{(I)}, \mathbf{C})$, $\ell_{ij} = h(\mathbf{m}, \mathbf{F}^{(ji)}, \mathbf{C})$, $\ell'_{ji} = h(\mathbf{m}+c_{ji}\boldsymbol{\delta}_i, \mathbf{F}, \mathbf{C}^{(ji)})$, and the vector $\mathbf{v}^{(i)}$ (respectively, the matrix $\mathbf{A}^{(ij)}$) is identical to the vector $\mathbf{v}$ (respectively, the matrix $\mathbf{A}$) but with its ith (respectively, ijth) component set to zero.

6.1.1 Correlated-failure–aware task reallocation policies

The general class of DTR policies presented in Section 4.1.1 is modified here to construct a correlated-failure–aware policy. First, it must be noticed from the definition of the general policy that when the estimate of the system load is accurate, the performance of the DTR policy is given by the reallocation criterion. A correlated-failure–aware DTR policy must account for the sources of correlation in the failure patterns. From the statistical analysis conducted in Chapter 5, the parameters r_v and $s_{u,v}$ are key in the generation of correlated failures; therefore, they must be included in the definition of the reallocation criterion.

With this, for spatially correlated failures, the following reallocation criterion is proposed as a mechanism to yield a correlated-failure–aware DTR policy:

$$\Lambda_j = \lambda_j \left(1 - \frac{\lambda_j^f}{\sum_{k\in V} \lambda_k^f}\right) r_j \left(\frac{1}{1+\sum_{v\in V} s_{j,v}}\right). \tag{6.3}$$

It becomes evident that the idea behind this definition is to favor the reallocation of tasks to those servers that are less vulnerable to fail, and simultaneously, to penalize the task reallocation to those servers having a strong interaction with other neighboring nodes. Note that the processing speed of the servers as well as the failure rate are still considered in the definition as in (4.14). It must be noticed also that when failures are uncorrelated all the $s_{i,j}$ parameters are zero and the proposed policy becomes proportional to (4.14), which was defined for the independent failure case. With this definition at hand, the optimization problem (4.5) can be solved to compensate for the effect of network delays.

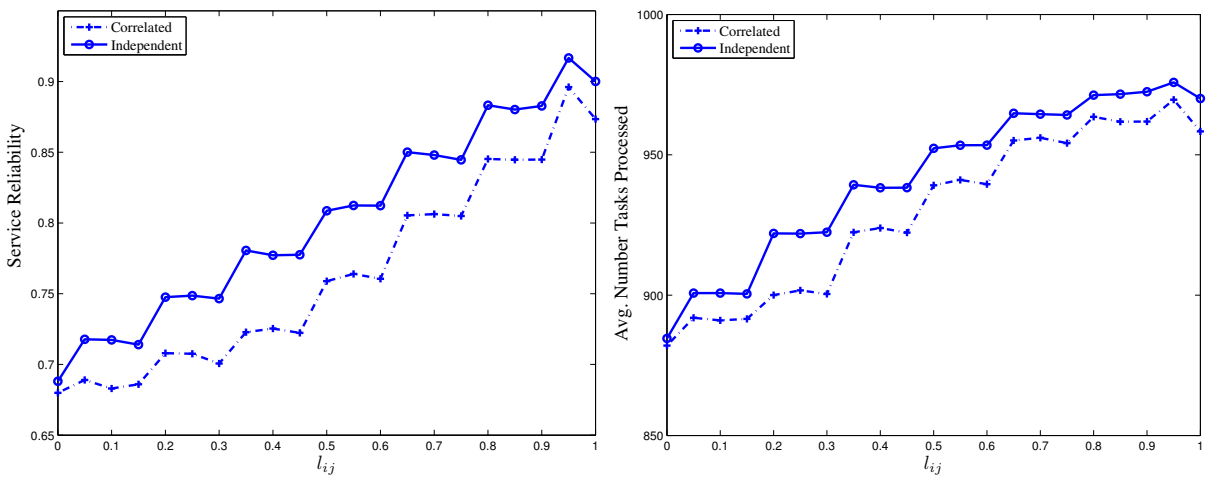
6.2 Assessing reliability in the presence of correlated failures

In order to assess the reliability of a DCS in the presence of correlated failures, the 17-server DCS has been considered. For simplicity of the presentation, in the calculations presented here two classes of servers have been considered: HP servers and standard servers. Since HP servers are expected to process more tasks than standard nodes, they are supplied with a larger number of logical connections. In particular, all those servers with four or more logical connections in the topology depicted in Fig. 5.1(c) are regarded as HP servers. The average task-processing time of the HP servers is 0.2 s, while the average task-processing times of the standard serves are 2, 3, and 4 s. An application composed of $M = 1000$ tasks is initially allocated uniformly onto the nodes.

In the failure more regarded here, conditional on a pattern of correlated failures, the average failure time of a cluster of failed servers is 3000 s. When independent failures are considered, the average failure time of a server is also 3000 s. Regarding task-transfer times, the mean transfer time of a group of tasks transferred over the network follows the first-oder approximation given in (2.1). The parameters of the Gibbs sampler have been adjusted to draw realizations of spatially correlated failures having the same average number of failed servers as those of an independent failure case. In addition, a DTR policy with a reallocation criterion based solely on the relative processing speed of the servers has been considered initially.

Figure 6.1(a) depicts the service reliability of the DCS as a function of the DTR policy. The DTR policy shown in the figure corresponds to the case of transferring tasks from a standard node to an HP server. It can be seen that, in spite of the average number of failures being the same, correlated failures reduce the service reliability as compared to the case of independent failures. For the case presented here,

the service reliability is reduced in approximately 2 to 6%. As a secondary metric of performance, the average number of tasks executed by the DCS before it fails has been estimated by means of MC simulations. Results are shown in Fig.6.1(b). It can be seen that, on average, a larger number of tasks are processed by the DCS when independent failures affect the behavior of the system. This result is expected because values of service reliability closer to one imply that a DCS is able to service the entire application more times before its fails.



(a) (b)

Figure 6.1: (a) Service reliability, and (b) average number of tasks served by the 17-server DCS as a function of fraction of tasks reallocated over the network.

The reduction in the service reliability, when correlated failures affect the DCS, is a consequence of disregarding the correlation information in the specification of the DTR policy. In order to see the effect on the reliability when such information is used in a policy, the DTR with a reallocation criterion based solely on the processing speed of servers is compared to the policy defined in (16). In this example, the workload is initially distributed uniformly over the standard servers only. Results on the comparison between the two DTR policies are shown in Fig. 6.2. it can be clearly observed that including the information about the spatial correlation in a

DTR policy yields a considerable increase in the service reliability as compared to a policy that omits such information. In this case an improvement of approximately 20% has been achieved.

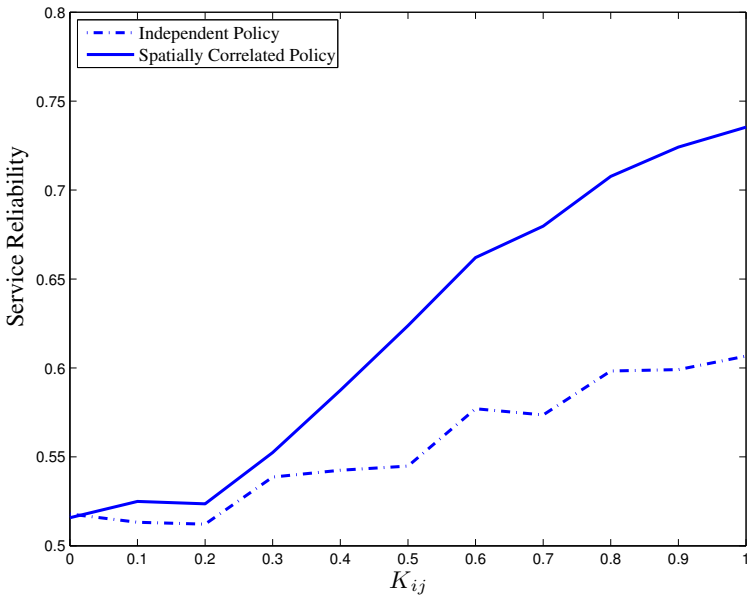


Figure 6.2: Service reliability as a function of the fraction of tasks reallocated .

6.3 Conclusions

The analytical model for service reliability has been extended to a setting where the failures at the servers are spatially correlated. The effects of such failures on the service reliability have been investigated. To this end, patterns of correlated-failure sampled from the MRF-based model developed in Chapter 5 are fed into the service reliability model to conditionally characterize the service reliability on a sample pattern of correlated failures. A comparison has been made between the service reliability of a DCS in presence of independent and correlated failures. Also, a correlated-failure–aware DTR policy has been proposed in order to enhance the

service reliability of the system.

Results show that correlated failures reduce both the service reliability and the average number of tasks executed by a DCS as compared to scenarios of independent failures. Moreover, the optimal selection of the number of tasks to reallocate among the servers in a DCS depends upon the degree of correlation in the failures. For example, results show that if one erroneously assume independent failures in the selection of the number of tasks to reallocate among the servers, the service reliability yielded by such reallocation is approximately 20% lower than that obtained when the correct correlation statistics of failures are employed. In addition, the average number of tasks executed by the DCS is reduced in approximately 6% as a consequence of the incorrect assumption on the degree of correlation in the failures.

Chapter 7

Future work

DTR policies must determine the amount of tasks to reallocate among the system servers. Such amount relies heavily on the estimate that a server has about the load at the remaining servers, and as a consequence, on the estimate of the total system workload. Inaccurate estimates of either system workload or server workload can lead to reduction in the performance of the DCS due to inefficient task exchange. In addition, an excessive amount of exchange of system information is not desirable due to communication as well as computational cost. Information fusion techniques, in conjunction with consensus theory, can be used in DCSs to yield accurate estimates of the system workload. In particular, the fact that consensus theory can construct consensed estimated values is a desirable property for estimating the system workload, because not only inaccurate state information can lead to a performance reduction, but also inconsistent state information does as stated in [25, 28]. As a future work, consensus theory can be employed as an information fusion technique to investigate the effect of both consistent yet inaccurate and inconsistent estimates of the system workload an their effect on the performance and reliability.

Also as a future work, the theory developed in these dissertation can be extended

to calculate other performance metrics, such as statistics of the queue-length of servers or jitter in the service time. Also, DTR can be studied in scenarios of green DC, where the goal of the DCS is to simultaneously reduce the average execution time and the power consumed by the servers in the DCS. Another avenue to explore is to extend the model to include the class of crash-recovery failures. Such extension, however, has the extra complication that increases the number of states visited by the process, and consequently, the computational costs are largely augmented.

Regarding the computational costs associated with the recursive characterization of the metrics, other approximation and dimension reduction techniques should be studied. For instance, the expression for the conditional probability of the regeneration event being a particular random time can be employed to discard those states with low likelihood. By doing so, the number of states visited at earlier recursions will be reduced, thereby reducing the total number of states visited as part of the evolution of the system dynamics. Another approach that can be investigated to reduce the state-space dimension is to exploit the vast amount of research on hybrid systems in control theory. For example, the concept of bisimulation seems to be a very useful tool to study systems with a large state space dimension, because by means of bisimulation one can create a subprocess with an equivalent state space of a desired hybrid process but with the advantage that bisimulation provides state space reduction.

Regarding the model for correlated failures, the theory can be extended to model more realistic correlated failures. For instance, failures induced by natural disasters or by attacks to the network infrastructure can take place at any location on the plane; therefore, a MRF-based model for correlated failures is not enough to abstract such situations. Point processes can be explored to generate the realizations of geographical failure centers. Then, such realizations can be employed to specify appropriated values for the robustness parameters in the MRF model for correlated

failures. The theory for correlated failures can be also extended to stochastically model the data dependencies generated by the class of parallel applications not considered in this work. In this scenarios, the graph modeling the data dependencies can be exploited to induce the logical data correlations. In a more practical approach, empirical data can be analyzed to refine the local specification proposed in this dissertation and construct a failure more driven by the type of applications analyzed. For instance, traces from distributed denial of service attacks or from disruptions due to large power outages can be employed as real-world examples.

Appendix A

Proof of Theorems 5 and 6

Consider that at time $\xi \geq 0$ before the synchronous DTR is performed by the servers, the initial system configuration is $\mathbf{S} = (\mathbf{m}, \mathbf{F}, \mathbf{0}, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$. Let us exploit the labeling mapping $h(\cdot)$ so that $\ell = h(\mathbf{m}, \mathbf{F}, \mathbf{0})$. The average execution time and the QoS in executing an application can be computed by conditioning on the regeneration time as follows:

$$\begin{aligned}
\overline{T}_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) &= \int_0^\infty \mathsf{E}[T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) | \tau_{\mathbf{a}} = \alpha] f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha \\
&= \int_0^\xi \mathsf{E}[T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) | \tau_{\mathbf{a}} = \alpha] f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha \\
&\quad + \int_\xi^\infty \mathsf{E}[T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) | \tau_{\mathbf{a}} = \alpha] f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha, \qquad \text{(A.1)} \\
Q_\ell(\xi, T_M, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) &= \int_0^\infty \mathsf{P}\{T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) < T_M | \tau_{\mathbf{a}} = \alpha\} f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha \\
&= \int_0^\xi \mathsf{P}\{T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) < T_M | \tau_{\mathbf{a}} = \alpha\} f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha \\
&\quad + \int_\xi^\infty \mathsf{P}\{T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) < T_M | \tau_{\mathbf{a}} = \alpha\} f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha. \qquad \text{(A.2)}
\end{aligned}$$

Note that the rightmost terms in (A.1) and (A.2) can be cast as:

$$\int_{\xi}^{\infty} \mathsf{E}[T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)|\tau_{\mathbf{a}}=\alpha] f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha$$

$$= \mathsf{E}[T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)|\tau_{\mathbf{a}} > \xi]\mathsf{P}\{\tau_{\mathbf{a}} > \xi\}, \tag{A.3}$$

$$\int_{\xi}^{\infty} \mathsf{P}\{T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) < T_M|\tau_{\mathbf{a}}=\alpha\} f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha$$

$$= \mathsf{P}\{T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) < T_M|\tau_{\mathbf{a}} > \xi\}\mathsf{P}\{\tau_{\mathbf{a}} > \xi\}. \tag{A.4}$$

To calculate the conditional expectations and the conditional probabilities in (A.1) to (A.4), the definition of the age-dependent regeneration time is exploited in conjunction with the law of total probabilities. Consequently, the conditional expectation as well as the conditional probability can be decomposed according to all possible, disjoint regeneration events occurring before the DTR action is performed. That is,

$$\mathsf{E}[T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)|\tau_{\mathbf{a}}=\alpha] = \sum_{k=1}^{n} \mathsf{P}\{\tau_{\mathbf{a}} = W_{k1} - a_{M_k}|\tau_{\mathbf{a}} = \alpha\}$$
$$\times \mathsf{E}[T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)|\tau_{\mathbf{a}}=\alpha, \tau_{\mathbf{a}}=W_{k1} - a_{M_k}]$$
$$\mathsf{P}\{T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)<T_M|\tau_{\mathbf{a}}=\alpha\} = \sum_{k=1}^{n} \mathsf{P}\{\tau_{\mathbf{a}} = W_{k1} - a_{M_k}|\tau_{\mathbf{a}} = \alpha\}$$
$$\times \mathsf{P}\{T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) < T_M|\tau_{\mathbf{a}}=\alpha, \tau_{\mathbf{a}}=W_{k1} - a_{M_k}\}$$
$$+ \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \mathsf{P}\{T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) <T_M|\tau_{\mathbf{a}}=\alpha, \tau_{\mathbf{a}}=X_{ij} - a_{F_{ij}}\}$$
$$\times \mathsf{P}\{\tau_{\mathbf{a}} = X_{ij} - a_{F_{ij}}|\tau_{\mathbf{a}} = \alpha\} + \sum_{k=1}^{n} \mathsf{P}\{\tau_{\mathbf{a}} = Y_k - a_{F_k}|\tau_{\mathbf{a}} = \alpha\}$$
$$\times \mathsf{P}\{T_\ell(\xi, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) < T_M|\tau_{\mathbf{a}}=\alpha, \tau_{\mathbf{a}}=Y_k - a_{F_k}\}. \tag{A.5}$$

Now, let $W_{ki}^{'}$, $Y_k^{'}$, $X_{jk}^{'}$, and $Z_{ik}^{'}$ denote all the random times associated with the DCS emerging at the regeneration time $\tau_{\mathbf{a}}$, all of them measured from α. Suppose now that the regeneration event happens to be the service of a task at the first server,

Appendix A. Proof of Theorems 5 and 6

$E_{\mathbf{a}} = \{\tau_{\mathbf{a}} = \alpha, \tau_{\mathbf{a}} = W_{11} - a_{M_1}\}$, and let $S_{\mathbf{a}} = \min\big(\min_{k,k\neq 1} W_{k1} - a_{M_k}, \min_k Y_k - a_{F_k}, \min_{j\neq k} X_{jk} - a_{F_{jk}}, \min_{k,i} Z_{ik} - a_{C_{ik}}\big)$. The joint distribution of a pair of random times associated with the emergent DCS, say $W_{k1}^{'} = W_{k1} - a_{M_k} - \tau_{\mathbf{a}}$ and $Y_k^{'} = Y_k - a_{F_k} - \tau_{\mathbf{a}}$, conditional on $E_{\mathbf{a}}$ is:

$$\begin{aligned}\mathsf{P}\{W_{k1}^{'} \leq t_1, Y_k^{'} \leq t_2 | E_{\mathbf{a}}\} &= \mathsf{P}\{W_{k1} \leq t_1 + a_{M_k} + \alpha, Y_k \leq t_2 + a_{F_k} + \alpha | E_{\mathbf{a}}\} \\ &= \mathsf{P}\{W_{k1} \leq t_1 + a_{M_k} + \alpha, Y_1 \leq t_2 + a_{F_k} + \alpha | D_{\mathbf{a}}\},\end{aligned}$$

where the last equation holds since $E_{\mathbf{a}}$ and $D_{\mathbf{a}} = \{W_{11} - a_{M_1} = \alpha, S_{\mathbf{a}} > \alpha\}$ are equivalent events. Exploiting Assumption A2 the conditional joint distribution can be written as

$$\begin{aligned}&\mathsf{P}\{W_{k1}^{'} \leq t_1, Y_k^{'} \leq t_2 | E_{\mathbf{a}}\} \\ &\quad= \mathsf{P}\{W_{k1} \leq t_1 + a_{M_k} + \alpha, Y_1 \leq t_2 + a_{F_k} + \alpha | W_{k1} > \alpha, Y_k > \alpha\} \\ &\quad= \frac{\mathsf{P}\{W_{k1} \leq t_1 + a_{M_k} + \alpha, Y_1 \leq t_2 + a_{F_k} + \alpha, W_{k1} > \alpha, Y_k > \alpha\}}{\mathsf{P}\{W_{k1} > \alpha, Y_k > \alpha\}} \\ &\quad= \frac{\mathsf{P}\{\alpha < W_{k1} \leq t_1 + a_{M_k} + \alpha\}}{\mathsf{P}\{W_{k1} > \alpha\}} \frac{\mathsf{P}\{\alpha < Y_k \leq t_2 + a_{F_k} + \alpha\}}{\mathsf{P}\{Y_k > \alpha\}} \\ &\quad= \mathsf{P}\{W_{k1}^{'} \leq t_1\}\mathsf{P}\{Y_{k1}^{'} \leq t_2\},\end{aligned}$$

where the last equation is obtained by replacing $W_{k1}^{'} = W_{k1} - a_{M_k} - \tau_{\mathbf{a}}$ and $Y_k^{'} = Y_k - a_{F_k} - \tau_{\mathbf{a}}$. Therefore, it can be concluded that conditional on the occurrence of a regeneration event, the random times emerging at the regeneration time are conditionally independent. Moreover, from the definitions of $W_{k1}^{'}$ and $Y_k^{'}$ it becomes clear that these emergent random times are aged versions, now further aged by s units of time, of the original random times W_{k1} and Y_k, respectively.

This conditional independence for the emerging DCS implies the following observations: (i) a fresh copy of the underlying stochastic process emerges at $\tau_{\mathbf{a}}$ but with a new initial age-dependent system-state matrix; (ii) the emergent stochastic process is independent of the original process and satisfies assumptions A1 and A2; and (iii) the independence of the new process allows to shift the time origin to $t = \alpha$. First, if

the regeneration event is the service of a task at the first server, after applying these observations the following holds

$$\begin{aligned}
&\mathsf{E}[T_\ell(\xi,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C)|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=W_{11}-a_{M_1}]\\
&\qquad=\mathsf{E}[\tau_{\mathbf{a}}+T_{\ell_1}(\xi,\mathbf{a}_M^{W_{11}},\mathbf{a}_F^{W_{11}},\mathbf{a}_C^{W_{11}})|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=W_{11}-a_{M_1}],\\
&\mathsf{P}\{T_\ell(\xi,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C)<T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=W_{11}-a_{M_1}\}\\
&\qquad=\mathsf{P}\{\tau_{\mathbf{a}}+T_{\ell_1}(\xi,\mathbf{a}_M^{W_{11}},\mathbf{a}_F^{W_{11}},\mathbf{a}_C^{W_{11}})<T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=W_{11}-a_{M_1}\},
\end{aligned}$$

where $T_{\ell_1}(\xi,\mathbf{a}_M^{W_{11}},\mathbf{a}_F^{W_{11}},\mathbf{a}_C^{W_{11}})$ is the random time taken by the DCS emerging at the regeneration time to serve all its tasks when the initial configuration is $\mathbf{S}^{W_{11}}=(\mathbf{m}^{W_{11}},\mathbf{F}^{W_{11}},\mathbf{C}^{W_{11}},\ \mathbf{a}_M^{W_{11}},\mathbf{a}_F^{W_{11}},\mathbf{a}_C^{W_{11}})$, where $\ell_1=h(\mathbf{m}^{W_{11}},\mathbf{F}^{W_{11}},\mathbf{C}^{W_{11}})$. This new initial configuration is precisely $\mathbf{m}^{W_{11}}=(r_1-1,\ r_2,\ldots,r_n)^T$, $\mathbf{F}^{W_{11}}=\mathbf{F}$, $\mathbf{C}^{W_{11}}=\mathbf{0}$, $\mathbf{a}_M^{W_{11}}=(0,\ a_{M_2}+s,\ldots,\ a_{M_2}+s)^T$, $\mathbf{a}_F^{W_{11}}=\mathbf{a}_F+s$, and $\mathbf{a}_C^{W_{11}}=\mathbf{a}_C+s$. By exploiting the independence between the emergent and the original process, and recalling that $\tau_{\mathbf{a}}=\alpha$, it is straightforward to obtain

$$\begin{aligned}
&\mathsf{E}[\tau_{\mathbf{a}}+T_{\ell_1}(\xi-\alpha,(\mathbf{a}_M+\alpha)^{(1)},\mathbf{a}_F+\alpha,\mathbf{a}_C+\alpha)|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=W_{11}-a_{M_1}]\\
&\quad=\mathsf{E}[\alpha|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=W_{11}-a_{M_1}]+\overline{T}_{\ell_1}(\xi-\alpha,(\mathbf{a}_M+\alpha)^{(1)},\mathbf{a}_F+\alpha,\mathbf{a}_C+\alpha)\\
&\quad=\alpha+\overline{T}_{\ell_1}(\xi-\alpha,(\mathbf{a}_M+\alpha)^{(1)},\mathbf{a}_F+\alpha,\mathbf{a}_C+\alpha),\qquad\text{(A.6)}\\
&\mathsf{P}\{\tau_{\mathbf{a}}+T_{\ell_1}(\xi-\alpha,(\mathbf{a}_M+\alpha)^{(1)},\mathbf{a}_F+\alpha,\mathbf{a}_C+\alpha)<T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=W_{11}-a_{M_1}\}\\
&\quad=\mathsf{P}\{T_{\ell_1}(\xi-\alpha,(\mathbf{a}_M+\alpha)^{(1)},\mathbf{a}_F+\alpha,\mathbf{a}_C+\alpha)<T_M-\alpha\}\\
&\quad=Q_{\ell_1}(\xi-\alpha,T_M-\alpha,(\mathbf{a}_M+\alpha)^{(1)},\mathbf{a}_F+\alpha,\mathbf{a}_C+\alpha).\qquad\text{(A.7)}
\end{aligned}$$

Second, if the regeneration event happens to be the arrival of a FN packet transferred from server i to server j, X_{ij}, the new system configuration is given by $\mathbf{S}^{X_{ij}}=(\mathbf{m}^{X_{ij}},\mathbf{F}^{X_{ij}},\mathbf{C}^{X_{ij}},\mathbf{a}_M^{X_{ij}},\mathbf{a}_F^{X_{ij}},\mathbf{a}_C^{X_{ij}})$, where $\mathbf{m}^{X_{ij}}=\mathbf{m}$, $\mathbf{F}^{X_{ij}}=\mathbf{F}^{(ji)}$, $\mathbf{C}^{X_{ij}}=\mathbf{0}$, $\mathbf{a}_M^{X_{ij}}=\mathbf{a}+\alpha$, $\mathbf{a}_F^{X_{ij}}=(\mathbf{a}_F+\alpha)^{ji}$, and $\mathbf{a}_C^{X_{ij}}=\mathbf{a}_C+\alpha$ because only the system-function state matrix is updated. Conducting the same analysis as for W_{11} it can be concluded

that

$$\begin{aligned}
&\mathsf{P}\{\tau_{\mathbf{a}}+T_{\ell_{ij}}(\xi-\alpha,\mathbf{a}_M+\alpha,(\mathbf{a}_F+\alpha)^{(ji)},\mathbf{a}_C+\alpha)<T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=X_{ij}-a_{F_{ij}}\}\\
&\quad=\mathsf{P}\{T_{\ell_{ij}}(\xi-\alpha,\mathbf{a}_M+\alpha,(\mathbf{a}_F+\alpha)^{(ji)},\mathbf{a}_C+\alpha)<T_M-s\}\\
&\quad=Q_{\ell_{ij}}(\xi-\alpha,T_M-\alpha,\mathbf{a}_M+\alpha,(\mathbf{a}_F+\alpha)^{(ji)},\mathbf{a}_C+\alpha),
\end{aligned}\tag{A.8}$$

where $\ell_{ij}=h(\mathbf{m},\mathbf{F}^{(ji)},\mathbf{0})$.

Third, if the regeneration event happens to be the failure of the j server, Y_j, the new system configuration is given by $\mathbf{S}^{Y_j}=(\mathbf{m}^{Y_j},\mathbf{F}^{Y_j},\mathbf{C}^{Y_j},\mathbf{a}_M^{Y_j},\,\mathbf{a}_F^{Y_j},\mathbf{a}_C^{Y_j})$, where $\mathbf{m}^{Y_j}=\mathbf{m}$, $\mathbf{F}^{Y_j}=\mathbf{F}^{(jj)}$, $\mathbf{C}^{Y_j}=\mathbf{0}$, $\mathbf{a}_M^{Y_j}=\mathbf{a}+\alpha$, $\mathbf{a}_F^{Y_j}=(\mathbf{a}_F+\alpha)^{(jj)}$, and $\mathbf{a}_C^{Y_j}=\mathbf{a}_C+\alpha$ because only the system-function state matrix is updated. Once again, conducting the same analysis as for W_{11} it can be concluded that

$$\begin{aligned}
&\mathsf{P}\{\tau_{\mathbf{a}}+T_{\ell'_j}(\xi-\alpha,\mathbf{a}_M+\alpha,(\mathbf{a}_F+\alpha)^{(jj)},\mathbf{a}_C+\alpha)<T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=Y_j-a_{F_{jj}}\}\\
&\quad=\mathsf{P}\{T_{\ell'_j}(\xi-\alpha,\mathbf{a}_M+\alpha,(\mathbf{a}_F+\alpha)^{(jj)},\mathbf{a}_C+\alpha)<T_M-\alpha\}\\
&\quad=Q_{\ell'_j}(\xi-\alpha,T_M-\alpha,\mathbf{a}_M+\alpha,(\mathbf{a}_F+\alpha)^{(jj)},\mathbf{a}_C+\alpha),,
\end{aligned}\tag{A.9}$$

where $\ell_{ij}=h(\mathbf{m},\mathbf{F}^{(jj)},\mathbf{0})$.

The first integral at the right-hand side of (3.9) is obtained by plugging (A.6) in (A.1), while the first three integrals at the right-hand side of (3.10) are obtained after plugging (A.7) to (A.9) in (A.2).

Similarly, the occurrence of the event $\{\tau_{\mathbf{a}}>\xi\}$ implies that, by definition of the regeneration time, nothing has changed in the system configuration before $t=\xi$. This means that the time origin can be shifted by ξ units of time, and this also means that, all the random times have further aged ξ units of time. This generates the terms at the rightmost part of (3.9) and (3.10).

In order to complete the characterization of the performance metrics, the term $G_X(s)$ must be completely specified. This means that formulae for calculating $f_{\tau_{\mathbf{a}}}(t)$,

the pdf of the age-dependent regeneration time, and $\mathsf{P}\{X=\tau_{\mathbf{a}}|\tau_{\mathbf{a}}=s\}$, the conditional probability of a particular random time being equal to the regeneration time, must be provided. For the sake of notation, let us index all the random times listed in Assumption A1 as well as their associated age parameters using the set $\mathcal{I}$. Let us also denote these random times as $T^{(j)}$ with $j \in \mathcal{I}$. Consider now the definition of the age-dependent regeneration time given in (3.5). From basic probability and from Assumption A2, the pdf of $\tau_{\mathbf{a}}$ can be computed in a straightforward manner using:

$$f_{\tau_{\mathbf{a}}}(t) = \sum_{j\in\mathcal{I}} f_{T^{(j)}|A_j}(t|a_j) \prod_{k\in\mathcal{I},k\neq j} \Big(1 - F_{T^{(k)}|A_k}(t|a_k)\Big), \tag{A.10}$$

where $A_j = \{V_j \geq a_j\}$ and $F_{T^{(j)}|A_j}(t|a_j) = F_{T^{(j)}}(t;a_j)$ [correspondingly, $f_{T^{(j)}|A_j}(t|a_j) = f_{T^{(j)}}(t;a_j)$] is the conditional distribution [correspondingly, the conditional density] that defines the cumulative distribution function [correspondingly, the pdf] of the aged version of the random time $T^{(j)}$ whose age parameter is a_j.

Finally, a formula for the conditional probability $\mathsf{P}\{T^{(j)}=\tau_{\mathbf{a}}|\tau_{\mathbf{a}}=s\}$ is obtained as follows. First note that the events $\{\tau_{\mathbf{a}} = T^{(j)}\}$ and $\{T^{(j)} < T^{(1)}, \ldots, T^{(j)} < T^{(j-1)}, T^{(j)} < T^{(j+1)}, \ldots, T^{(j)} < T^{(p)}\}$ are equivalent, where p is the cardinality of the set $\mathcal{I}$. Thus,

$$\begin{aligned}\mathsf{P}\{T^{(j)}=\tau_{\mathbf{a}}|\tau_{\mathbf{a}}=s\} =& \mathsf{P}\{T^{(j)} < T^{(1)}, \ldots, T^{(j)} < T^{(j-1)}, T^{(j)} < T^{(j+1)}, \ldots, \\ & T^{(j)} < T^{(p)}|\tau_{\mathbf{a}}=s\}.\end{aligned}$$

Next, we note that the event $\{\tau_{\mathbf{a}} = s\}$ implies that s units of time have elapsed since all the random times have been fired and compete to be the regeneration event. Consequently, all the random times have aged s units of time in addition to the ages specified in $\mathbf{a}_0$. Thus, we can write

$$\begin{aligned}\mathsf{P}\{T^{(j)}=\tau_{\mathbf{a}}|\tau_{\mathbf{a}}=s\} =& \mathsf{P}\{T^{(j)}_{a_j+s} < T^{(1)}_{a_1+s}, \ldots, T^{(j)}_{a_j+s} < T^{(j-1)}_{a_{j-1}+s}, T^{(j)}_{a_j+s} < T^{(j+1)}_{a_{j+1}+s}, \ldots, \\ & T^{(j)}_{a_j+s} < T^{(p)}_{a_p+s}|\tau_{\mathbf{a}}=s\},\end{aligned}$$

where $T^{(j)}_{a_j+s}$ is the aged version the random time $T^{(j)}$. With this the condition on the event $\{\tau_{\mathbf{a}} = s\}$ can be dropped, provided that the event defining the age of the jth random time is redefined as $A'_j = \{V_j \geq a_j + s\}$. By invoking the law of total probability and by exploiting Assumption A2 it is straightforward to obtain the following relationship:

$$\mathsf{P}\{T^{(j)} = \tau_{\mathbf{a}} | \tau_{\mathbf{a}} = s\} = \int_0^\infty f_{T^{(j)}|A'_j}(t; a_j + s) \prod_{k \in \mathcal{I}, k \neq j} \big(1 - F_{T^{(k)}|A'_j}(t|a_k + s)\big) ds. \tag{A.11}$$

□

Appendix B

Sketch of proof of Theorems 7 and 8

Consider that after the DTR action has been performed, the initial system configuration is $\mathbf{S} = (\mathbf{m}, \mathbf{F}, \mathbf{C}, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$. The average execution time and the QoS in executing an application can be computed by conditioning on the regeneration time as follows:

$$\overline{T}_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) = \int_0^\infty \mathsf{E}[T_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) | \tau_{\mathbf{a}} = \alpha] f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha, \tag{B.1}$$

$$Q_\ell(0, T_M, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) = \int_0^\infty \mathsf{P}\{T_\ell(0, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C) < T_M | \tau_{\mathbf{a}} = \alpha\} f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha. \tag{B.2}$$

The conditional expectations and the conditional probabilities in (B.1) and (B.2) can be computed from the definition of the age-dependent regeneration time in conjunction with the law of total probabilities by conditioning on all the possible, disjoint

regeneration events:

$$\mathsf{E}[T_\ell(0,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C)|\tau_{\mathbf{a}}=\alpha] = \sum_{k=1}^{n}\mathsf{P}\{\tau_{\mathbf{a}} = W_{k1} - a_{M_k}|\tau_{\mathbf{a}} = \alpha\}$$
$$\times\,\mathsf{E}[T_\ell(0,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C)|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=W_{k1}-a_{M_k}] + \sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}$$
$$\mathsf{P}\{\tau_{\mathbf{a}} = Z_{ji} - a_{C_{ji}}|\tau_{\mathbf{a}} = \alpha\}\mathsf{E}[T_\ell(0,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C)|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=Z_{ji}-a_{C_{ji}}] \quad \text{(B.3)}$$

$$\mathsf{P}\{T_\ell(0,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C)<T_M|\tau_{\mathbf{a}}=\alpha\} = \sum_{k=1}^{n}\mathsf{P}\{\tau_{\mathbf{a}} = W_{k1} - a_{M_k}|\tau_{\mathbf{a}} = \alpha\}$$
$$\times\,\mathsf{P}\{T_\ell(0,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C) < T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=W_{k1}-a_{M_k}\}$$
$$+\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\mathsf{P}\{T_\ell(0,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C)<T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=X_{ij}-a_{F_{ij}}\}$$
$$\times\,\mathsf{P}\{\tau_{\mathbf{a}} = X_{ij} - a_{F_{ij}}|\tau_{\mathbf{a}} = \alpha\} + \sum_{k=1}^{n}\mathsf{P}\{\tau_{\mathbf{a}} = Y_k - a_{F_k}|\tau_{\mathbf{a}} = \alpha\}$$
$$\times\,\mathsf{P}\{T_\ell(0,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C) < T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=Y_k-a_{F_k}\}$$
$$+\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\mathsf{P}\{T_\ell(0,\mathbf{a}_M,\mathbf{a}_F,\mathbf{a}_C)<T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=Z_{ji}-a_{C_{ji}}\}$$
$$\times\,\mathsf{P}\{\tau_{\mathbf{a}} = Z_{ji} - a_{C_{ji}}|\tau_{\mathbf{a}} = \alpha\}. \quad \text{(B.4)}$$

Note now that, as in the case of the analysis conducted for the DCS emerging before t_b, conditional on the occurrence of a regeneration event, the random times emerging at the regeneration time are conditionally independent. Therefore, the conditional independence for the emerging DCS implies the following observations: (i) a fresh copy of the underlying stochastic process emerges at $\tau_{\mathbf{a}}$ but with a new initial age-dependent system-state matrix; (ii) the emergent stochastic process is independent of the original process and satisfies assumptions A1 and A2; and (iii) the independence of the new process allows to shift the time origin to $t = s$. Since in Appendix A, these observations were applied to all the regeneration events but those triggered by the arrival of tasks from server i to server j, it suffices to show

here the effect of such regeneration event on the system configuration. Thus, if the regeneration event happens to be Z_{ij}, the new system configuration is given by $\mathbf{S}^{Z_{ij}} = (\mathbf{m}^{Z_{ij}}, \mathbf{F}^{Z_{ij}}, \mathbf{C}^{X_{ij}}, \mathbf{a}_M^{Z_{ij}}, \mathbf{a}_F^{Z_{ij}}, \mathbf{a}_C^{Z_{ij}})$, where $\mathbf{m}^{Z_{ij}} = \mathbf{m} + l_{ij}\boldsymbol{\delta}_j$, $\mathbf{F}^{Z_{ij}} = \mathbf{F}$, $\mathbf{C}^{Z_{ij}} = \mathbf{C}^{ij}$, $\mathbf{a}_M^{Z_{ji}} = \mathbf{a} + \alpha$, $\mathbf{a}_F^{Z_{ij}} = \mathbf{a}_F + \alpha$, and $\mathbf{a}_C^{Z_{ij}} = (\mathbf{a}_C + \alpha)^{ij}$ because both system-queue state vector and the network state matrix are updated upon the arrival of tasks at a server. Conducting the same analysis as for W_{11} it can be concluded that

$$\begin{aligned}
&\mathsf{E}[\tau_{\mathbf{a}} + T_{\ell'_{ij}}(0, \mathbf{a}_M + \alpha, \mathbf{a}_F + \alpha, (\mathbf{a}_C + \alpha)^{(ij)}) | \tau_{\mathbf{a}} = \alpha, \tau_{\mathbf{a}} = Z_{ij} - a_{C_{ij}}] \\
&\quad = \mathsf{E}[\alpha | \tau_{\mathbf{a}} = \alpha, \tau_{\mathbf{a}} = Z_{ij} - a_{C_{ij}}] + \overline{T}_{\ell'_{ij}}(0, \mathbf{a}_M + \alpha, \mathbf{a}_F + \alpha, (\mathbf{a}_C + \alpha)^{(ij)}) \\
&\quad = \alpha + \overline{T}_{\ell'_{ij}}(0, \mathbf{a}_M + \alpha, \mathbf{a}_F + \alpha, (\mathbf{a}_C + \alpha)^{(ij)}) \qquad \text{(B.5)} \\
&\mathsf{P}\{\tau_{\mathbf{a}} + T_{\ell'_{ij}}(0, \mathbf{a}_M + \alpha, \mathbf{a}_F + \alpha, (\mathbf{a}_C + \alpha)^{(ij)}) < T_M | \tau_{\mathbf{a}} = \alpha, \tau_{\mathbf{a}} = Z_{ij} - a_{C_{ij}}\} \\
&\quad = \mathsf{P}\{T_{\ell'_{ij}}(0, \mathbf{a}_M + \alpha, \mathbf{a}_F + \alpha, (\mathbf{a}_C + \alpha)^{(ij)}) < T_M - \alpha\} \\
&\quad = Q_{\ell'_{ij}}(0, T_M - \alpha, \mathbf{a}_M + \alpha, \mathbf{a}_F + \alpha, (\mathbf{a}_C + \alpha)^{(ij)}), \qquad \text{(B.6)}
\end{aligned}$$

where $\ell'_{ij} = h(\mathbf{m} + l_{ij}\boldsymbol{\delta}_j, \mathbf{F}, \mathbf{C}^{ij})$. The two integrals at the right-hand side of (3.9) is obtained by plugging (A.6) and (B.5) in (B.1), while the $\mathsf{E}[\tau_{\mathbf{a}}]$ term appears from (B.5) after some algebra and the application of the law of total probability. The four integrals at the right-hand side of (3.10) are obtained after plugging in (A.7) to (A.9) and (B.6) in (B.2).

Finally, formulae for calculating $f_{\tau_{\mathbf{a}}}(t)$, the pdf of the age-dependent regeneration time, and $\mathsf{P}\{X = \tau_{\mathbf{a}} | \tau_{\mathbf{a}} = s\}$, the conditional probability of a particular random time being equal to the regeneration time are provided by (A.10) and (A.11). □

Appendix C

Sketch of proof of Theorems 9 to 12

Consider that the initial system configuration is $\widetilde{\mathbf{S}} = (\mathbf{m}, \mathbf{f}, \mathbf{c}, \mathbf{a}_M, \mathbf{a}_F, \mathbf{a}_C)$. The average approximated execution time and the approximated QoS in executing an application can be computed by conditioning on the regeneration time as follows:

$$\overline{T}_\ell(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C) = \int_0^\infty \mathsf{E}[T_\ell(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C) | \tau_{\mathbf{a}} = \alpha] f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha, \tag{C.1}$$

$$Q_\ell(0, T_M, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C) = \int_0^\infty \mathsf{P}\{T_\ell(0, \widetilde{\mathbf{a}}_M, \widetilde{\mathbf{a}}_F, \widetilde{\mathbf{a}}_C) < T_M | \tau_{\mathbf{a}} = \alpha\} f_{\tau_{\mathbf{a}}}(\alpha)\, d\alpha. \tag{C.2}$$

Proceeding in the same manner as in the previous cases, the conditional expectations and the conditional probabilities in (B.1) and (B.2) can be computed from the definition of the age-dependent regeneration time in conjunction with the law of

total probabilities by conditioning on all the possible, disjoint regeneration events:

$$\begin{aligned}
\mathsf{E}[T_\ell(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C)|\tau_{\mathbf{a}}=\alpha] &= \sum_{k=1}^{n}\mathsf{P}\{\tau_{\mathbf{a}}=W_{k1}-a_{M_k}|\tau_{\mathbf{a}}=\alpha\}\\
&\times \mathsf{E}[T_\ell(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C)|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=W_{k1}-a_{M_k}]+\sum_{i=1}^{n}\\
&\mathsf{P}\{\tau_{\mathbf{a}}=\widetilde{Z}_j-\widetilde{a}_{C_i}|\tau_{\mathbf{a}}=\alpha\}\mathsf{E}[T_\ell(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C)|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=\widetilde{Z}_j-\widetilde{a}_{C_j}] \qquad \text{(C.3)}\\
\mathsf{P}\{T_\ell(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C)<T_M|\tau_{\mathbf{a}}=\alpha\} &= \sum_{k=1}^{n}\mathsf{P}\{\tau_{\mathbf{a}}=W_{k1}-a_{M_k}|\tau_{\mathbf{a}}=\alpha\}\\
&\times \mathsf{P}\{T_\ell(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C)<T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=W_{k1}-a_{M_k}\}\\
&+\sum_{k=1}^{n}\mathsf{P}\{\tau_{\mathbf{a}}=Y_k-a_{F_k}|\tau_{\mathbf{a}}=\alpha\}\mathsf{P}\{T_\ell(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C)<T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=Y_k-a_{F_k}\}\\
&+\sum_{i=1}^{n}\mathsf{P}\{T_\ell(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C)<T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=\widetilde{Z}_j-\widetilde{a}_{C_{ji}}\}\\
&\times \mathsf{P}\{\tau_{\mathbf{a}}=\widetilde{Z}_j-\widetilde{a}_{C_j}|\tau_{\mathbf{a}}=\alpha\}. \qquad \text{(C.4)}
\end{aligned}$$

Applying the same principles as in the proof of Theorems 5 to 10, conditional on the occurrence of a regeneration event, the approximated random times emerging at the regeneration time are conditionally independent. Therefore, the conditional independence for the emerging DCS implies the following observations: (i) a fresh copy of the underlying stochastic process emerges at $\tau_{\mathbf{a}}$ but with a new initial age-dependent system-state matrix; (ii) the emergent stochastic process is independent of the original process and satisfies assumptions A1 and A2; and (iii) the independence of the new process allows to shift the time origin to $t = s$. In particular, if the regeneration event happens to be the simultaneous arrival of $l_j = \sum_{i=1}^{n} l_{ij}$ tasks to the jth server, $\widetilde{Z}_j$, the new system configuration is given by $\mathbf{S}^{\widetilde{Z}_j} = (\mathbf{m}^{\widetilde{Z}_j}, \mathbf{f}^{\widetilde{Z}_j}, \mathbf{c}^{\widetilde{Z}_j}, \mathbf{a}_M^{\widetilde{Z}_j}, \mathbf{a}_F^{\widetilde{Z}_j}, \mathbf{a}_C^{\widetilde{Z}_j})$, where $\mathbf{m}^{\widetilde{Z}_j} = \mathbf{m} + l_j\boldsymbol{\delta}_j$, $\mathbf{f}^{\widetilde{Z}_j} = \mathbf{f}$, $\mathbf{c}^{\widetilde{Z}_j} = \mathbf{c}^{(i)}$, $\mathbf{a}_M^{\widetilde{Z}_j} = \mathbf{a} + \alpha$, $\mathbf{a}_F^{\widetilde{Z}_j} = \mathbf{a}_F + \alpha$, and $\mathbf{a}_C^{\widetilde{Z}_j} = (\mathbf{a}_C + \alpha)^i$ because both system-queue state vector and the network state vector are updated upon the simultaneous arrival of tasks at a server. Thus, it can be

concluded that

$$\begin{aligned}
&\mathsf{E}[\tau_{\mathbf{a}}+T_{\ell'_j}(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C)|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=\widetilde{Z}_j-\widetilde{a}_{C_j}]\\
&\quad=\mathsf{E}[\alpha|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=\widetilde{Z}_j-\widetilde{a}_{C_{ij}}]+T_{\ell'_j}(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C)\\
&\quad=\alpha+T_{\ell'_j}(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C), \qquad \text{(C.5)}\\
&\mathsf{P}\{\tau_{\mathbf{a}}+T_{\ell'_j}(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C)<T_M|\tau_{\mathbf{a}}=\alpha,\tau_{\mathbf{a}}=\widetilde{Z}_j-\widetilde{a}_{C_j}\}\\
&\quad=\mathsf{P}\{T_{\ell'_j}(0,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C)<T_M-s\}\\
&\quad=Q_{\ell'_j}(0,T_M-\alpha,\widetilde{\mathbf{a}}_M,\widetilde{\mathbf{a}}_F,\widetilde{\mathbf{a}}_C), \qquad \text{(C.6)}
\end{aligned}$$

where $\ell'_j=\widetilde{h}(\mathbf{m}+l_j\boldsymbol{\delta}_j,\mathbf{f},\mathbf{c}^{(i)})$. The two integrals at the right-hand side of (3.12) is obtained by plugging (A.6) and (C.5) in (C.1), while the $\mathsf{E}[\tau_{\mathbf{a}}]$ term appears from (C.5) after some algebra and the application of the law of total probability. The three integrals at the right-hand side of (3.14) are obtained after plugging in (A.7), (A.9) and (C.6) in (C.2).

Finally, formulae for calculating $f_{\tau_{\mathbf{a}}}(t)$, the pdf of the age-dependent regeneration time, and $\mathsf{P}\{X=\tau_{\mathbf{a}}|\tau_{\mathbf{a}}=s\}$, the conditional probability of a particular random time being equal to the regeneration time are provided by (A.10) and (A.11). □

Bibliography

[1] The Load Balancing Group. http://www.ece.unm.edu/lb, 2007.

[2] C. T. Abdallah, N. Alluri, J.D. Birdwell, J. Chiasson, V. Chupryna, and T. Wang. Z. Tang. A linear time delay model for studying load balancing instabilities in parallel computations. *International Journal Systems Sciences*, 2003.

[3] C. T. Abdallah, J. D. Birdwell, J. Chiasson, V. Chupryna, Z. Tang, and T. Wang. The effect of time delays in the stability of load balancing algorithms for parallel computations. *Automatica*, 2002.

[4] J. Al-Jaroodi, N. Mohamed, H. Jiang, and D. Swanson. Modeling parallel applications performance on heterogeneous systems. In *Proc. Int. Parallel and Distributed Processing Symposium (IPDPS)*, page 160.2, Washington, DC, USA, 2003. IEEE Computer Society.

[5] S. Ali, H. J. Siegel, M. Maheswaran, S. Ali, and D. Hensgen. Task execution time modeling for heterogeneous computing systems. In *HCW '00: Proceedings of the 9th Heterogeneous Computing Workshop*, page 185, Washington, DC, USA, 2000. IEEE Computer Society.

[6] G. Attiya and Y. Hamam. Reliability oriented task allocation in heterogeneous distributed computing systems. In *Proc. Ninth Int. Symp. Computers and Comms.*, pages 68–73, 2004.

[7] G. Attiya and Y. Hamam. Task allocation for maximizing reliability of distributed systems: a simulated annealing approach. *Journal Parallel and Dist. Computing*, 66:1259–1266, 2006.

[8] A. Aubry, A. Rossi, M.-L. Espinouse, and M. Jacomino. Minimizing setup costs for parallel multi-purpose machines under load-balancing constraint. *European Journal of Operational Research*, 187(3):1115–1125, 2008.

[9] A. Bobbio, A. Puliafito, M. Scarpa, and M. Telek. Webspn: A web-accessible petri net tool. In *Proc. Conference on Web-based Modeling & Simulation*, 1998.

[10] A. Bobbio, A. Puliafito, and M. Tekel. A modeling framework to implement preemption policies in non-markovian SPNs. *IEEE Trans. Software Engineering*, 26(1):36–54, 2000.

[11] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi. *Queueing Networks and Markov Chains*. John Wiley & Sons, Inc., second edition, 2006.

[12] M. Bouissou and J.-L. Bonc. A new formalism that combines advantages of fault-trees and markov models: Boolean logic driven markov processes. *Reliability Engineering and System Safety*, 82(2):149–163, 2003.

[13] P. Bremaud. *Markov chains, Gibbs fields, Monte Carlo simulation, and queues*. Springer-Verlag, New York, second edition, 2001.

[14] M. L. Bujorianu, J. Lygeros, and M. C. Bujorianu. Bisimulation for general stochastic hybrid systems. In R. Alur and G. J. Pappas, editors, *Hybrid Systems: Computation and Control*, volume 3414 of *Lecture Notes in Computer Science*, pages 198–214. Springer Verlag, Berlin, 2005.

[15] T. L. Casavant and J. G. Kuhl. A taxonomy of scheduling in general-purpose distributed computing systems. *IEEE Transactions on Software Engineering*, 1988.

[16] U. V. Catalyurek, E. G. Boman, K. D. Devine, D. Bozdag, R. T. Heaphy, and L.A. Riesen. Hypergraph-based dynamic load balancing for adaptive scientific computations. In *Proceedings of 21st International Parallel and Distributed Processing Symposium (IPDPS'07)*. IEEE, 2007.

[17] C.-I.H. Chen. Task allocation and reallocation for fault tolerance in multicomputer systems. *Trans. Aerospace & Electronic Syst.*, 30:1094–1104, 1994.

[18] J. Chiasson, Z. Tang, J. Ghanem, C. T. Abdallah, J. D. Birdwell, M. M. Hayat, and H. Jerez. The effect of time delays on the stability of load balancing algorithms for parallel computations. *IEEE Transactions on Control Systems Technology*, 2005.

[19] The Marine Corps. Joint Warning and Reporting Network (JWARN). http://www.globalsecurity.org/military/library/budget/fy2001/dot-e/other/01jwarn.html, Last accessed, August 2010.

[20] D. R. Cox. The analysis of non-markovian stochastic processes by the inclusion of supplementary variables. *Proceedings of the Cambridge Philosophical Society*, 51:433–441, 1965.

[21] G. Cybenko. Dynamic load balancing for distributed memory multiprocessors. *IEEE Transactions on Parallel and Distributed Computing*, 1989.

[22] Y-S. Dai and G. Levitin. Optimal resource allocation for maximizing performance and reliability in tree-structured grid services. *IEEE Trans. Reliability*, 56:444–453, 2007.

[23] Y-S. Dai, G. Levitin, and K.S. Trivedi. Performance and reliability of tree-structured grid services considering data dependence and failure correlation. *IEEE Transactions on Computers*, 56:925–936, 2007.

[24] Y.S. Dai, M. Xie, and K.L. Poh. Modeling and analysis of correlated software failures of multiple types. *IEEE Trans. Reliability*, 54:100–106, 2005.

[25] S. Dhakal. *Load Balancing in Communication Constrained Distributed Systems: A Probabilistic Approach.* PhD thesis, University of New Mexico, 2006.

[26] S. Dhakal, M. M. Hayat, J. E. Pezoa, C. T. Abdallah, J. D. Birdwell, and J. Chiasson. Load balancing in the presence of random node failure and recovery. In *Proc. IEEE IPDPS '06*, Rhodes, Greece, 2006.

[27] S. Dhakal, M. M. Hayat, J. E. Pezoa, C. T. Abdallah, J. D. Birdwell, and J. N. Chiasson. Load balancing in the presence of random node failure and recovery. In *Proceedings of the 20th International Parallel and Distributed Processing Symposium (IPDPS 2006)*, 2006.

[28] S. Dhakal, M. M. Hayat, J. E. Pezoa, C. Yang, and D. A. Bader. Dynamic load balancing in distributed systems in the presence of delays: A regeneration-theory approach. *IEEE Trans. Parallel and Dist. Systems*, 18:485–497, 2007.

[29] S. Dhakal, B.S. Paskaleva, M. M. Hayat, E. Schamiloglu, and C. T. Abdallah. Dynamical discrete-time load balancing in distributed systems in the presence of time delays. In *Proc. IEEE CDC '03*, Maui, HI, 2003.

[30] S. Dhakal, J. E. Pezoa, and M. M. Hayat. A regeneration-based approach for resource allocation in cooperative distributed systems. In *Proc. ICASSP*, pages III–1261–III–1264, 2007.

[31] M. Dodge and R. Kitchin. *The Atlas of Cyberspace.* Addison Wesley, 2008.

[32] A. Dogan and F. Özgüner. Biobjective scheduling algorithms for execution time–reliability trade-off in heterogeneous computing systems. *The Computer Journal*, 48:300–314, 2005.

[33] C. Fetzer. Perfect failure detection in timed asynchronous systems. *IEEE Trans. on Computers*, 52(2):99–112, 2003.

[34] L. Fiondella and S. S. Gokhale. Estimating system reliability with correlated component failures. *International Journal of Reliability and Safety*, 4(2–3):188–205, 2010.

[35] S. Fu and C. Z. Xu. Quantifying temporal and spatial correlation of failure events for proactive management. In *Proc. 26th IEEE International Symp. on Reliab. Dist. Systems*, 2007.

[36] Mark Galassi, Jim Davies, James Theiler, Brian Gough, Gerard Jungman, Michael Booth, and Fabrice Rossi. *Gnu Scientific Library: Reference Manual*. Network Theory Ltd., 2003.

[37] R. G. Gallager. Basic limits on protocol information in data communication networks. *IEEE Transactions on Information Theory*, 1976.

[38] R. German. Non-Markovian analysis. In *Lectures on formal methods and performance analysis: first EEF/Euro summer school on trends in computer science*, pages 156–182, New York, NY, USA, 2002. Springer-Verlag New York, Inc.

[39] J. Ghanem. *Implementation of Load-Balancing Policies in Distributed Systems*. PhD thesis, University of New Mexico, 2004.

[40] J. Ghanem, C. T. Abdallah, M. M. Hayat, S. Dhakal, J.D Birdwell, J. Chiasson, and Z. Tang. Implementation of load balancing algorithms over a local area network and the internet. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, 2004.

[41] K. Goseva-Popstojanova and K. S. Trivedi. Failure correlation in software reliability model. *IEEE Trans. Reliability*, 49:37–48, 2000.

[42] Serge Haddad and Patrice Moreaux. Approximate analysis of non-markovian stochastic systems with multiple time scale delays. In *Proc. IEEE MASCOTS '04*, pages 23–30, Washington, DC, USA, 2004.

[43] Y. Hamam and K.S. Hindi. Assignment of program tasks to processors: a simulated annealing approach. *J. Op. Research*, 122, 2000.

[44] M. M. Hayat, S. Dhakal, C. T. Abdallah, J. D. Birdwell, and J. Chiasson. *Dynamic time delay models for load balancing. Part II: Stochastic analysis of the effect of delay uncertainty*, chapter Advances in Time Delay Systems, pages 355–368. LNCS v. 38. Springer-Verlag, 2004.

[45] M. M. Hayat, J. E. Pezoa, D. Dietz, and S. Dhakal. *Wiley Handbook of Science and Technology for Homeland Security*, volume 4, chapter Dynamic load balancing for robust distributed computing in the presence of topological impairments. Wiley, 2009.

[46] B. Hendrickson. Load balancing fictions, falsehoods and falacies. *Applied Mathematical Modelling*, 2000.

[47] L. E. Holloway, B. H. Krogh, and A. Giua. A survey of petri net methods for controlled discrete event systems. *Discrete Event Dynamic Systems*, 7(2):151–190, 1997.

[48] C. Hui and S.T. Chanson. Hydrodynamic load balancing. *IEEE Trans. Parallel and Dist. Systems*, 10:1118–1137, 1999.

[49] K. Jagannathan, E. Modiano, and L. Zheng. Effective resource allocation in a queue: How much control is necessary? In *46th Annual Allerton Conference on Communication, Control, and Computing*, pages 508–515, 2008.

[50] G. Jiang and G. Cybenko. Temporal and spatial distributed event correlation for network security. In *Proc. of American Control Conference*, Boston, MA, 2004.

[51] C. Kelling. Timenet-sim-a parallel simulator for stochastic petri nets. In *Proc. 28th Annual Simulation Symposium*, 1995.

[52] M. C. Kim and P. H. Seong. Reliability graph with general gates: An intuitive and practical method for system reliability analysis. *Reliability Engineering and System Safety*, 78(3):239–246, 2002.

[53] R. Kinderman and J. L. Snell. *Markov Random Fields and Their Applications*. AMS, Providence, 1980.

[54] G. Koole, P. Sparaggis, and D. Towsley. Minimizing response times and queue lengths in systems of parallel queues. *J. Applied Probability*, 36:1185–1193, 1999.

[55] Z. Lan, V.E. Taylor, and G. Bryan. Dynamic load balancing for adaptive mesh refinement application. In *Proc. ICPP 01*, Valencia, Spain, 2001.

[56] H. Lee, S. Chin, J. Lee, D. Lee, K. Chung, S. Jung, and H. Yu. A resource manager for optimal resource selection and fault tolerance service in grids. In *Proc. Int. Symp. Cluster Comp. and Grid*, Chicago, IL, 2004.

[57] O. Lee, S. Lee, S. Kim, and I. Chung. An efficient load balancing algorithm employing a symmetric balanced incomplete block design. In *Computational Science–ICCS*, pages 147–154, 2003.

[58] G. Levitin and Y.-S.Dai. Service reliability and performance in grid system with star topology. *Reliability Engineering and System Safety*, 92:40–46, 2007.

[59] S. Z. Li. *Markov Random Field Modeling in Image Analysis*. Springer-Verlag, London, 2009.

[60] C.-J. Liao and Y.-C. Chung. Tree-based parallel load-balancing methods for solution-adaptive finite element graphs on distributed memory multicomputers. *IEEE Trans. Parallel Distrib. Syst.*, 1999.

[61] M. Litzkow, M. Livny, and M. Mutka. Condor - a hunter of idle workstations. In *Proc. 8th Int. Conf. Dist. Comp. Systems*, pages 104–111, 1988.

[62] D. Logothetis, V. Mainkar, and K. Trivedi. Transient analysis of non-markovian queues via markov regenerative processes. In *Probability and Statistics – A. J. Medhi Festschrift*, pages 109–131, 1996.

[63] T. Ma, J. Hillston, and S. Anderson. Evaluation of the qos of crash-recovery failure detection. In *Proceedings of the ACM symposium on Applied computing*, pages 538–542. ACM, 2007.

[64] M. G. Markakis, E. H. Modiano, and J. N. Tsitsiklis. Scheduling policies for single-hop networks with heavy-tailed traffic. In *Forty-Seventh Annual Allerton Conference Allerton House, UIUC, Illinois, USA*, pages 112–120, 2009.

[65] J. Palmer and I. Mitrani. Empirical and analytical evaluation of systems with multiple unreliable servers. In *Proc. Int. Conf. Dependable Syst. and Nets.*, pages 517–525, Philadelphia, PA, 2006.

[66] J. E. Pezoa, S. Dhakal, and M. M. Hayat. Decentralized load balancing for improving reliability in heterogeneous distributed systems. In *In Proc. ICPP 09*, Vienna, Austria, 2009.

[67] J. E. Pezoa, S. Dhakal, and M. M. Hayat. Maximizing service reliability in distributed computing systems with random failures: Theory and implementation. *IEEE Trans. Parallel and Dist. Systems*, 21(10):1531–1544, 2010.

[68] J. E. Pezoa, M. M. Hayat, and M. Rahnamay-Naeini. Task reallocation for reliable distributed computing in the presence of spatially correlated failures. *To be submitted to IEEE Trans. Parallel and Dist. Systems*, 2010.

[69] J. E. Pezoa, M. M. Hayat, and Z. Wang. Analytical characterization of performance and reliability in non-markovian heterogeneous distributed computing systems. *To be submitted to IEEE Trans. Parallel and Dist. Systems*, 2010.

[70] J. E. Pezoa, M. M. Hayat, Z. Wang, and S. Dhakal. Optimal task reallocation in heterogeneous distributed computing systems with age-dependent delay statistics. In *Proc. ICPP 2010*, San Diego, CA, USA, 2010.

[71] S. Pllana, I. Brandic, and S. Benkner. Performance modeling and prediction of parallel and distributed computing systems: A survey of the state of the art. In *CISIS '07: Proc. First International Conference on Complex, Intelligent and Software Intensive Systems*, pages 279–284, Washington, DC, USA, 2007.

[72] V. Ravi, B.S.N. Murty, and J. Reddy. Nonequilibrium simulated-annealing algorithm applied to reliability optimization of complex systems. *IEEE Trans. Reliability*, 46:233–239, 1997.

[73] R. Shah, B. Veeravalli, and M. Misra. On the design of adaptive and decentralized load balancing algorithms with load estimation for computational grid environments. *IEEE Trans. Parallel and Dist. Systems*, 18:1675–1686, 2007.

[74] S.M. Shatz and J-P. Wang. Models & algorithms for reliability-oriented task-allocation in redundant distributed-computer systems. *IEEE Trans. Reliability*, 38:16–27, 1989.

[75] R. Sheahan, L. Lipsky, and P. Fiorini. The effect of different failure recovery procedures on the distribution of task completion times. In *Proc. IEEE DPDNS '05*, Denver, CO, 2005.

[76] V. Shestak, E. K. P. Chong, A. A. Maciejewski, and H. J. Siegel. Robust sequential resource allocation in heterogeneous distributed systems with random compute node failures. In *Proc. Int. Parallel and Distributed Processing Symposium (IPDPS)*, Roma, Italy, 2009. IEEE Computer Society.

[77] V. Shestak, J. Smith, A.A. Maciejewski, and H.J. Siegel. Stochastic robustness metric and its use for static resource allocations. *Journal Parallel Dist. Computing*, 68:1157–1173, 2008.

[78] J. Sonnek, A. Chandra, and J. B. Weissman. Adaptive reputation-based scheduling on unreliable distributed infrastructures. *IEEE Trans. Parallel and Dist. Systems*, 18(11):1551–1564, 2007.

[79] S. Srinivasan and N.K. Jha. Safety and reliability driven task allocation in distributed systems. *IEEE Trans. Parallel and Dist. Systems*, 10:238–251, 1999.

[80] S. A. Taheri. *Cross Layer Design of Communication Network Layers.* PhD thesis, University of New Mexico, Albuquerque, NM, USA., 2004.

[81] D. Tang and R. K. Iyer. Analysis and modeling of correlated failures in multicomputer systems. *IEEE Trans. Comput.*, 41(5):567–577, 1992.

[82] Z. Tang, J. D. Birdwell, J. Chiasson, C. T. Abdallah, and M. M. Hayat. Resource-constrained load balancing controller for a parallel database. *IEEE Trans. on Control Systems Technology*, 16(4):834–840, 2008.

[83] X. Tanga, K. Li, R. Li, and B. Veeravalli. Reliability-aware scheduling strategy for heterogeneous distributed computing systems. *J. Parallel Distrib. Comput.*, 70(11):941–952, 2010.

[84] M. Trehel, C. Balayer, and A. Alloui. Modeling load balancing inside groups using queuing theory. In *Proc. 10th Int. Conf. on Parallel and Distributed Computing System*, New Orleans, LO, 1997.

[85] K. S. Trivedi, A. Bobbio, G. Ciardo, R. German, A. Puliafito, and M. Telek. Non-markovian petri nets. In *Proc. 1995 ACM SIGMETRICS*, pages 263–264, New York, NY, USA, 1995. ACM.

[86] The University of New Mexico. The UNM Center for Advanced Research Computing (CARC). http://www.carc.unm.edu, Last accessed, August 2010.

[87] A. J. van der Schaft. Bisimulation of dynamical systems. In R. Alur and G. J. Pappas, editors, *Hybrid Systems: Computation and Control*, volume 2993 of *Lecture Notes in Computer Science*, pages 555–569. Springer Verlag, Berlin, 2004.

[88] D.P. Vidyarthi and A.K. Tripathi. Maximizing reliability of a distributed computing system with task allocation using simple genetic algorithm. *Journal of Systems Architecture*, 47:549–554, 2001.

[89] C. Xu, B. Monien, R. Lüling, and F. C. M. Lau. Nearest neighbor algorithms for load balancing in parallel computers. *Concurrency: Practice and Experience*, 1997.