10-2010

Maximizing Service Reliability in Distributed Computing Systems with Random Node Failures: Theory and Implementation

Jorge E. Pezoa

Sagar Dhakal

Majeed M. Hayat

IEEE Transactions on Parallel and Distributed Systems Vol. 21, No.10 (October, 2010): 1531 – 1544. DOI.

Maximizing Service Reliability in Distributed Computing Systems with Random Node Failures: Theory and Implementation

Jorge E. Pezoa
University of New Mexico, Albuquerque, NM
Sagar Dhakal
Naval Research Laboratory, Washington DC
Majeed M. Hayat
University of New Mexico, Albuquerque, NM

Abstract:

In distributed computing systems (DCSs) where server nodes can fail permanently with nonzero probability, the system performance can be assessed by means of the service reliability, defined as the probability of serving all the tasks queued in the DCS before all the nodes fail. This paper presents a rigorous probabilistic framework to analytically characterize the service reliability of a DCS in the presence of communication

uncertainties and stochastic topological changes due to node deletions. The framework considers a system composed of heterogeneous nodes with stochastic service and failure times and a communication network imposing random tangible delays. The framework also permits arbitrarily specified, distributed load-balancing actions to be taken by the individual nodes in order to improve the service reliability. The presented analysis is based upon a novel use of the concept of stochastic regeneration, which is exploited to derive a system of difference-differential equations characterizing the service reliability. The theory is further utilized to optimize certain load-balancing policies for maximal service reliability; the optimization is carried out by means of an algorithm that scales linearly with the number of nodes in the system. The analytical model is validated using both Monte Carlo simulations and experimental data collected from a DCS testbed.

SECTION 1 Introduction

A distributed computing system (DCS) allows its users to process large, time-consuming workloads in a cooperative fashion. To achieve this goal, each workload has to be divided into smaller and independent units, called tasks. Next, these tasks have to be redistributed to appropriate computational elements, where they are concurrently processed. Tasks have to be intelligently allocated onto the nodes in order to efficiently use the resources available in the system. Such task allocation is referred to in the literature as *load balancing* (LB). LB is of great importance in distributed computing since, as commonly known, the performance of a given DCS strongly depends upon the distribution of the tasks in the system [1]. Furthermore, the LB problem belongs to a more general class of problems in resource allocation. These problems appear not only in distributed computing but also in routing in wireless networks, telecommunications, data replication in hard-drive arrays, and other problems in computer science and operational research [2], [3], [4], [5], [6].

LB policies rely on the effective exchange of load state information among the nodes. This information is used to estimate whether nodes are imbalanced or not with respect to other nodes in the DCS. Moreover, load state information is utilized to calculate both the appropriate amount of tasks that needs to be reallocated to other nodes and the appropriate set of nodes receiving the load. When the communication network imposes stochastic, tangible delays, the load state information available to the nodes may be severely dated and therefore misleading. Moreover, such delays automatically imply that the effect of task reallocation is not instantaneous. Clearly, it is expected that the success of any scheduler to balance the workload is degraded by communication limitations [7], [8].

The dynamics of DCS becomes further complicated in volatile or harsh environments in which nodes are prone to fail permanently (as in scenarios where massive disruptions result from weapons of mass destruction). In such cases, messages have to be broadcasted among working nodes in order to detect and isolate faulty nodes. Once again, due to network stochastic communication delays, information available to each node about the number of the functional nodes in the DCS may not be current; as such, LB policies as well as methods for reallocating tasks originally assigned to faulty nodes must be analyzed employing a probabilistic framework.

The role of LB 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 workload [1], [8], the probability of successfully serving an entire workload [9], [10], [11], [12], [13], [14], [15], the probability of serving a workload within a given amount of time [16], the average queue length of a node [17], [18], and the total sum of communication and service times [4], [19]. In addition, the problem of LB has been studied

under both static and dynamic scenarios. In static LB, a centralized entity allocates the tasks offline, that is, tasks are allocated prior to their execution in the DCS [10], [11], [17]. In contrast, in dynamic LB, tasks are queued at the nodes and LB is triggered online whenever there is an imbalance in the DCS [7], [8], [19], [20].

LB has been effectively employed to reduce the effect of node failures on the execution of a workload. The objective is to maximize the service reliability, while the response time of the workload is simultaneously minimized. To date, existing analytical solutions to this problem have been based upon multiobjective optimization approaches. Some approaches have assumed deterministic communication delays [21], [22], [23], [24] while introducing task and/or hardware redundancy to compensate for the delays [25], [26]. Other solutions either exploit a priori information on the network configuration [27] or provide computationally fast solutions by using heuristic algorithms such as genetic algorithms [28] and simulated annealing [29], [30]. Most relevant to this paper are the recent works by Dai et al. [10], [11]. The authors solve the static LB 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 an earlier work, we have studied the effect of node failure and recovery on the average response time of a workload served by a two-node DCS [9].

In this paper, we consider the problem of LB for maximizing the service reliability of a DCS. Unlike Dai et al., we address the dynamic LB problem and propose an online decentralized solution. We extend the model presented in [15] and characterize the service reliability of DCSs composed of an arbitrary number of nodes. Further, in this paper, we characterize the dynamics of the service reliability as a function of the balancing instant. Note that due to communication limitations, there is a trade-off between having accurate account of the node states prior to LB by means of delaying the LB and immediacy of LB action (*to* prevent wasting time). This new view of reliability offered by our analytical approach enables us to optimally select when the balancing action should be taken. Potential applications of this work include resilient distributed computing for battlefield management systems (as distributed computing is performed in harsh environments where nodes can fail permanently), grid computing (where nodes can leave the DCS at any time), and wireless sensor networks in harsh environments.

This paper is organized as follows. In Section 2, we build the regeneration-based stochastic theory for analyzing the reliability of DCSs. In Section 3, we apply the theory to devise LB strategies that maximize the reliability of a DCS. In the same section, the analytical model for reliability is validated and the performance of the LB strategies is tested, both theoretically and experimentally. Our conclusions are presented in Section 4.

SECTION 2 Theory

2.1 Problem Statement

Consider a DCS composed of n nodes communicating over a fully connected network. Consider also that a workload comprising M independent, indivisible tasks has to be processed by the system. Suppose that the service time of a task at each node is random and suppose also that server nodes can fail permanently at any random time. Assume that at $t = 0$ all the nodes are functioning and tasks are allocated on the nodes so that the jth node has in its queue m_j tasks, with $\sum_{j=1}^{n} m_j = M$. The problem addressed in this paper is concerned with maximizing the service reliability of the workload, i.e., maximizing the probability of serving all the tasks before all nodes fail.

In order to maximize the service reliability, LB is performed at time $t_b \geq 0$ so that each functional node, the jth node, say, transfers a positive amount, L_{jk}, of tasks to the kth node, with $j \neq k$, which is functioning according to the knowledge of the jth node. Naturally, these task exchanges over the network take random transfer times. Additionally, we have assumed that, at $t = 0$, each node broadcasts a QI packet that takes a random amount of time to reach the destination nodes.

The dynamics of the DCS are governed by the random times associated to the service of tasks, the failure of nodes, and the transfer time of both information and tasks in the network. These random times are important in our analysis and are defined next.

2.1.1 Definitions and Assumptions

Let the random variable W_{ki} be the service time of the ith task at the kth node, and let X_{jk}^{Q} be the transfer time of the QI packet sent from the jth to the kth node, $j \neq k$. The failure time of the kth node is represented by the random variable Y_k, and the transfer time of the failure-notice (FN) packet sent from the jth to the kth node is represented by the random variable $X_{jk}^{F}(j \neq k)$. Finally, let the random variable Z_{ik} be the transfer time of the ith group of tasks sent to the kth node. We require the following assumptions on these random variables:

Assumption A1

Assumption A1 Exponential distribution of the random times

The random variables W_{ki}, X_{ik}^{Q}, Y_k, and X_{jk}^{F} *follow exponential distributions with rates* $\lambda_{d_k}, \lambda_{jk}^{Q}, \lambda_{f_k}$ and λ_{jk}^{F}, *respectively. The random variable* Z_{ik} *is assumed to follow an exponential distribution conditional on the number of tasks transferred to the* k *th node*.

Assumption A2

Assumption A2 Independence of the random times

All the random variables listed in Assumption A1 are mutually independent.

Assumptions on the exponential distribution of the service and failure times are commonly adopted in the literature [1], [10], [17], [29]. Regarding the transfer times, our assumptions are justified according to our prior work [8], [9], [15] and the empirical data obtained from the experiments conducted over the distributed computing architecture to be discussed in Section 3. In addition, we have assumed that the mean transfer time of the ith group of tasks being transferred to the kth node follows the first-order approximation: $\mathrm{E}[Z_{i,k}] = \tilde{\lambda}_{i,k}^{-1} = a_{jk} l_{ik} + b_{jk}$, where a_{jk} and b_{jk} are positive constants (in seconds per task and seconds, respectively) that depend upon the communication channel connecting the jth and the kth nodes, and l_{ik} is the number of tasks in the ith group. This first-order approximation captures the linear dependence of the mean transfer time on: 1) the number of tasks to be transferred; 2) the end-to-end transmission time per task, through the parameter a_{jk} that is related to the bandwidth; and 3) the combined effects of the absolute minimum end-to-end propagation time and any delays resulting from queuing (due to congestion), which can be represented by a single parameter, b_{jk}.

Our analysis focuses on characterizing and maximizing the service reliability when the DCS is dedicated to a specific user, i.e., we consider the reliability question one workload at a time. To this end, we assume in our analysis that there are no future arrivals of *external tasks* to the DCS after the submission of a workload at $t = 0$. To tackle the reliability problem in the more general, shared setting, where workloads arrive continuously, the analysis presented here must be modified to distinguish between the different workloads in the system, and in addition, queuing disciplines (related to workload prioritization) have to be considered. However, the method presented here is an upper bound for such a general setting with continuous workload arrivals and it gives the *maximum* reliability that the DCS can guarantee to an individual user. Finally, it must be remarked that the

theory presented here assumes the existence of a completely reliable fault-tolerance mechanism. When such ideal mechanisms are not available, the failure of a node can produce task losses. Consequently, in this new setting, the service reliability has to be defined as the probability of serving all the tasks initially allocated to the nodes.

2.2 Task Reallocation Policy

In order to maximize the service reliability of the DCS, each functioning node executes a distributed, albeit synchronous, LB policy at $t = t_b$. The execution of the workload can be accomplished successfully only if task redundancy is provided by the DCS. Task redundancy is provided here by means of a trivial backup policy that is executed only in the event of node failure. The backup policy is asynchronous and it is triggered either at the actual failure instant of the nodes or at the reception of tasks by the backup system of a failed node.

2.2.1 Distributed Load-Balancing Policy

First, since the dynamic LB policy executed by the nodes is distributed, each node must determine independently the total amount of tasks to reallocate to other nodes. At the balancing instant, $t_b \geq 0$, the jth functioning node computes its excess load by comparing its local load to the estimated average load in the system. Let $Q_j(t_b)$be the number of tasks queued at the jth node at time t_b. Also, let $\hat{Q}_{\ell,j}(t_b)$ be the estimate of the number of tasks queued at the ℓth functioning node as perceived by the jth node at time t_b, with $\ell \neq j$. Here, we assume that $\hat{Q}_{\ell,j}(t_b) = m_\ell$if the QI packet has been received by the jth node at the time t_b and $\hat{Q}_{\ell,j}(t_b) = 0$ otherwise. The excess load of the jth node at time t_b is defined as

$$L_j^{ex}(t_b) \triangleq Q_j(t_b) - \frac{\Lambda_j}{\sum_{\ell \in \mathcal{W}_j} \Lambda_\ell} \hat{M}_j(t_b), \quad (1)$$

where $\hat{M}_j(t_b) = Q_j(t_b) + \sum_{\ell=1,\ell\neq j}^{n} \hat{Q}_{\ell,j}(t_b)$ is the estimate of the workload in the system as perceived by the jth node at time $t = t_b$, $\mathcal{W}_j$is the collection of nodes that are functioning as perceived by the jth node at time $t = t_b$. Note that in order to accurately estimate the initial workload of the system, nodes have to consider the queue length information received from all the nodes, not only the information from the functioning nodes. Finally, the Λ_j**s** are parameters that can be defined in several ways in order to establish different balancing criteria. For example, if the Λ_j**s** are associated with the processing speed, namely, $\Lambda_j = \lambda_{d_j}$, then the imbalance in the DCS is determined by the relative computing powers of the nodes. Alternatively, if the Λ_js are associated to the reliability of the nodes, namely, $\Lambda_j = \lambda_{f_j}^{-1}$, then the reliability of the nodes determines the amount of imbalance. Yet another option is to define theΛ_js so that we simultaneously transfer fewer tasks to the less-reliable nodes and transfer larger number of tasks to the faster processors. With this criterion in mind, we can define $\Lambda_j = \lambda_{d_j}\left(1 - \frac{\lambda_{f_j}}{\sum_{k\in\mathcal{W}_i}\lambda_{f_k}}\right)$Note that in the case of an extremely reliable node $(\lambda_f \approx 0)$, the parameter Λ_j is approximately equal to the processing rate of the node. On the contrary, for an unreliable node, the parameter Λ_j is only a reduced fraction of its processing rate.

Second, each node has to determine the amount of tasks to reallocate to the remaining nodes in the system. Let us define the collection $\mathcal{V}$ of overloaded nodes in the DCS as all those nodes that, at the balancing instant, perceive themselves as overloaded with respect to their perceived fair share of the total workload of the system. Mathematically, we define $\mathcal{V} \triangleq \{j: L_j^{ex}(t_b) > 0\}$. Similarly, for each overloaded node j, we define the collection $\mathcal{U}_j$ of candidate task-receiver nodes as all those nodes that, at time t_b, are perceived by the sender

node 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 node as perceived by the jth node and is defined as $L_{k,j}^{ex}(t_b) \triangleq \hat{Q}_{k,j}(t_b) - \Lambda_k \hat{M}_j(t_b) / \sum_{\ell \in \mathcal{W}_i} \Lambda_\ell$.

Third, the jth node partitions its excess load among all the candidate task-receiver nodes. For the kth candidate task-receiver node, the partition p_{jk} is defined as $p_{jk} \triangleq L_{k,j}^{ex}(t_b) / \sum_{\ell \in \mathcal{U}_i} L_{\ell,j}^{ex}(t_b)$ whenever $k \in \mathcal{U}_j$. For convenience, the partition $p_{ji} = 0$ for all $i \notin \mathcal{U}_j$. Note that in the special case when the task transfer times are negligible, the partitions p_{jk} will maximize the service reliability under all node-failure rates [8]. This is due to the fact that upon the occurrence of a failure, the unserved tasks of the failed node can instantly join the queues of other surviving nodes. In general, however, the above partitions p_{jk} may not be effective and must be adjusted in order to compensate for the effects of the random transfer times. The load to be migrated from the jth to the kth must be adjusted according to what is called the *load-balancing gain*[8], [9], [15], [20], which is denoted as K_{jk}, yielding $L_{jk}(t_b) = \lfloor K_{jk} p_{jk} L_j^{ex}(t_b) \rfloor$. $\lfloor x \rfloor$is the greatest integer less than or equal to x.)

Note that at the balancing instant, the excess load at node j as well as the partitions p_{jk} are fixed quantities; the LB policy is determined by the LB gains. Here, LB gains are regarded as parameters that have to be optimally selected in order to maximize the service reliability. Given that the quantity $p_{jk} L_j^{ex}$ defines the maximum number of tasks to be exchanged from node j to k, we have assumed here that the LB gains are rational numbers in the interval [0, 1]. Finally, we arrange in matrix form the LB gains with the convention that $K_{jj} = 0$ for all j. We denote such matrix by $\mathbf{K}$. From this, the LB policy $\mathbf{K}$ refers to a task reallocation policy specified by the LB gains K_{ij} and executed at $t = t_b$.

2.2.2 Task Recovery in the Event of Node Failure

The reliability problem tackled here can be solved only if the DCS provides task redundancy. Task redundancy is provided by means of a backup system that is attached to each node. This mechanism for task redundancy is a distributed version of the centralized method described in [12]. It must be noted that the backup system does not service any tasks. More specifically, in the event of node failure, the backup system 1) broadcasts an FN packet to alert the nodes about the change in the number of functioning nodes; 2) reallocates all the unfinished tasks among those nodes perceived to be functioning; and 3) handles the reception of tasks that were in transit to the jth node before its failure, and next, reallocates the received tasks among the functioning nodes. In particular, if the jth node has failed, its backup equipment reallocates L_{jk}^F tasks to the kth node, with $k \in \mathcal{W}_j$. In order to simplify the work of the backup system, the number of tasks L_{jk}^F is computed using the formula $L_{jk}^F = \lfloor Q_j \Lambda_k \left(\sum_{l \in \mathcal{W}_j} \Lambda_\ell \right)^{-1} \rfloor$.

The remainder of this section focuses on deriving recurrence equations that characterize the service reliability. We begin by introducing some necessary definitions of key system variables.

2.3 State Model for the Service Reliability

2.3.1 System Queue, System Function, and Network State

At any time, the configuration of a DCS can be described using the following quantities: 1) the number of tasks queued at each node; 2) the functional or dysfunctional state of each node in the system; and 3) the amount of tasks in transit over the communication network. In what follows, we formally develop the necessary notation to describe the time-varying DCS configuration.

Recall that $Q_i(t)$denotes the queue length of the ith node in the DCS at time t. For $i \neq j$, we use the binary variable $q_{ij}(t)$ to indicate if the ith node is informed (“1”) or not (“0”) about the queue length of the jth node.

That is, the $q_{ij}(t)$variable describes if the QI packet broadcasted by the jth node has been received or not by the jth node. We can arrange the $Q_i(t)$and $q_{ij}(t)$ variables in an n-by-n matrix, denoted by $\mathbf{Q}(t)$ whose ith diagonal element contains $Q_i(t)$ and its ijth off-diagonal element contains the $q_{ij}(t)$variables. We term the Q(t) matrix as the *system-queue state*. For example, in a two-node DCS, the matrix

$$\mathbf{Q}(t_0) = \begin{pmatrix} m_1 & 0 \\ 1 & m_2 \end{pmatrix}$$

at time $t = t_0$ corresponds to the configuration for which the first node has in its queue m_1 tasks and is uninformed about the queue length of node 2, while node 2 has m_2 tasks in its queue and is informed about the number of tasks queued at node 1 at $t = 0$.

Let $f_i(t)$ be a binary variable representing the working ("1") or failed ("0") state of the ith node at time t. For $i \neq j$, we define $f_{ij}(t) = 1$ (correspondingly, $f_{ij}(t) = 0$) to indicate that the jth node is functioning (correspondingly, faulty) as perceived by the ith node at time t. As in the case of the system queue state, we arrange all these variables in an n-by-–n matrix and introduce the *system function state*, which is denoted by the matrix $\mathbf{F}(t)$. Note that as in the case of the queue length information, the random transfer time of FN packets introduces uncertainty on the functioning state that a node perceives about the other nodes in the DCS.

In addition, due to stochastic transfer times in the communication network, each group of tasks being migrated over the network has a random transfer time. Let the nonnegative integer $g_k(t)$ represent the number of different groups of tasks that are in transit, from different nodes, to the kth node at time t. Let al.o l_{ik} be the number of tasks in the ith group being transferred to the kth node. For convenience of notation, we can assign the vector $\mathbf{c}_k(t)$ to the kth node such that the first component of $\mathbf{c}_k(t)$ is always set to $g_k(t)$, while its remaining components are set to l_{ik}. More precisely, $\mathbf{c}_k(t) \triangleq \left(g_k(t) l_{1k} l_{2k} \dots l_{gk(t)k}\right)$. We now define the *network state* as the concatenated vector $\mathbf{C}(t) \triangleq \left(\mathbf{c}_1(t), \dots, \mathbf{c}_n(t)\right)$. For example, in a three-node DCS, the vector $\mathbf{C}(t_0) = ([2101], [1\ 5], [0])$ at $t = t_0$ corresponds to a network state for which two different groups of tasks (10 tasks in the first group and 1 task in the second group) are being transferred to the first node $(\mathbf{c}_1(t_0) = [2101])$one group of five tasks is being transferred to the second node $(\mathbf{c}_2(t_0) = [15])$,and there are no tasks in transit to the third node $(\mathbf{c}_3(t_0) = [0])$.

2.3.2 Service Reliability

At this point, we are ready to define formally the service reliability of a DCS. Let $T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0)$ denote the random time taken by the DCS to serve its entire workload if the LB denoted by $\mathbf{K}$ is performed by all functioning nodes at time t_b, and the initial system configuration at $t = 0$ is as specified by $\mathbf{Q}_0 = \mathbf{Q}(0), \mathbf{F}_0 = \mathbf{F}(0)$. More precisely, we define $T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) \triangleq \inf\{t > 0$:diag$(\mathbf{Q}(t)) = 0$ and $\mathbf{C}(t) = 0\}$, where diag$(\mathbf{Q})$ is a vector formed by all the elements in the diagonal of the $\mathbf{Q}$ matrix. Note that by construction, the workload completion time is infinite when all the nodes have failed and at least one task remains unserved. Note also that P$\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) = \infty\} > 0$ since servers can fail permanently with nonzero probability. Our objective is to calculate the service reliability that is defined as the probability that all the tasks can be served before all servers fail, that is $R_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) \triangleq$ P$\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty\}$. Note that the service reliability is less than unity since P$\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) = \infty\} > 0$.

2.4 Regeneration Time

The main idea of our analysis is to introduce a *regeneration event*, and analyze the queuing system emerging immediately after the occurrence of the regeneration event. The key property of the regeneration event is that upon its occurrence, a *fresh* copy of the original stochastic process (from which the random variable $T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0)$ is defined) will emerge, nonetheless having a *new* initial system configuration that transpires from the regeneration event. To this end, we introduce the *regeneration time*, τ, which is the

minimum of the following five random variables: the time to the *first* task service by any node, the time to the *first* occurrence of failure at any node, the time to the *first* arrival of a QI packet at any node, the time to the *first* arrival of an FN packet at any node, or the time to the *first* arrival of a group of tasks at any node. More precisely, $\tau \triangleq \min\left(\min_k(W_{k1}), \min_{j\neq k}(X_{jk}^Q), \min_k(Y_k), \min_{j\neq k}(X_{jk}^F), \min_{k,i}(Z_{ki})\right)$.

Suppose that the initial system state is described by $\mathbf{Q}_0, \mathbf{F}_0$, and $\mathbf{C}_0$. The occurrence of the regeneration event $\{\tau = s\}$ gives birth to a new DCS at $\tau = s$ whose random times satisfy Assumptions AI and A2 while having its own initial system configuration. The new initial system configuration can be either one of the following:

1. a new initial task distribution when the regeneration event is a service to a task at a node;
2. a new system queue state when the regeneration event is the reception of a QI packet;
3. a new initial task distribution, a new system function state, and a new network state when the regeneration event is a node failure;
4. a new system queue state and a new system function state when the regeneration event is the reception of an FN packet; or
5. a new initial task distribution and a new network state when the regeneration event is the reception of a group of tasks by a node.

2.5 Characterization of the Service Reliability

Our main results are given in Theorems 1 and 2. Theorem 1 characterizes the service reliability of an n-node DCS in the form of a difference-differential equation. Theorem 2provides the initial condition required to solve Theorem 1.

We will introduce necessary notation that will facilitate keeping track of the changes in the initial system configuration. While the notation may seem cumbersome, it is extremely effective in allowing us to write equations in Theorems 1 and 2 compactly. Let δ_{ij} denote an n-by-−n matrix with all its entries equal to zero except that its ijth element is equal to 1. Let $\mathbf{A}$ be a matrix. We denote by $\mathbf{A}^{ij}$ a matrix that is identical to $\mathbf{A}$ but with its ijth component set to zero. Also, recall that f_{ii} is the ith diagonal element of $\mathbf{F}_0$, $L_{ik}^F$is the number of tasks reallocated from the ith to the kth node upon failure of node i, l_{ji} is the number of tasks in the jth group in transit to the ith node, and $\mathbf{c}_k \triangleq \left(g_k l_{1k} l_{2k} \dots l_{g_k k}\right)$ is the vector representing the number of tasks in transit to the kth node at a certain time t. Vectors $\mathbf{C}_0^{Y_i}$ and $\mathbf{C}_0^{Z_{ij}}$ represent the change in the network state when the ith node fails and when it receives the jth group of tasks, respectively. More precisely, vectors $\mathbf{C}_0^{Y_i}$ and $\mathbf{C}_0^{Z_{ji}}$ are defined as $\mathbf{C}_0^{Y_i} \triangleq \left(\mathbf{c}_1^{Y_i}, \dots, \mathbf{c}_i, \dots, \mathbf{c}_n^{Y_i}\right)$ with $\mathbf{c}_k^{Y_i} = \left(g_k + u\left(L_{ik}^F\right) l_{1k} \dots l_{g,k} L_{ik}^F\right)$, $u(\cdot)$ the unit-step function, and $\mathbf{C}_0^{Z_{ji}} = \left(\mathbf{c}_1^{Z_{ji}}, \dots, \mathbf{c}_i^{Z_{ji}}, \dots, \mathbf{c}_n^{Z_{ji}}\right)$ with $\mathbf{c}_i^{Z_{ji}} = \left(q_i - 1 l_{1i} \dots l_{(j-1)i} l_{(j+1)i} \dots l_{g_k i}\right)$ and $\mathbf{c}_k^{Z_{ji}} = \left(g_k + u\left(L_{ik}^F\right) l_{1k} \dots l_{g_k k} L_{ik}^F\right)$ for $k \neq i$.

Theorem 1

Consider an n*-node DCS with an arbitrarily specified initial system configuration* $\mathbf{Q}_0 = \mathbf{Q}(0), \mathbf{F}_0 = \mathbf{F}(0), \mathbf{C}_0 = \mathbf{C}(0)$. *The service reliability satisfies the difference-differential equation*:

$$\frac{d}{dt_b} R_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) = \sum_{i=1}^{n} \lambda_{d_i} R_{\mathbf{K}}(t_b; \mathbf{Q}_0 - \delta_{ii}, \mathbf{F}_0, \mathbf{C}_0)$$
$$+ \sum_{i=1}^{n} \sum_{i=1, j\neq i}^{n} \lambda_{ij}^{Q} R_{\mathbf{K}}(t_b; \mathbf{Q}_0 + \delta_{ji}, \mathbf{F}_0, \mathbf{C}_0)$$
$$+ \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \lambda_{ij}^{F} R_{\mathbf{K}}(t_b; \mathbf{Q}_0^{ji}, \mathbf{F}_0^{ji}, \mathbf{C}_0)$$
$$+ \sum_{i=1}^{n} \sum_{j=1}^{g_i} \tilde{\lambda}_{j,i} R_{\mathbf{K}}\left(t_{b;} \mathbf{Q}_0 + f_{ii} l_{ji} \delta_{ii}, \mathbf{F}_0, \mathbf{C}_0^{Z_{ij}}\right)$$
$$+ \sum_{i=1}^{n} \lambda_{f_i} R_{\mathbf{K}}(t_b; \mathbf{Q}_0^{ii}, \mathbf{F}_0^{ii}, \mathbf{C}_0^{Y_i}) - \lambda R_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0),$$
$$with \lambda = \sum_{i=1}^{n} \left(\lambda_{d_i} + \lambda_{f_i} + \sum_{j=1}^{g_i} \tilde{\lambda}_{j,i}\right) + \Sigma \sum_{i,j=1, j\neq i}^{n} \left(\lambda_{ij}^{Q} + \lambda_{ij}^{F}\right).$$

Proof

See Appendix. □

It must be noted that the characterization for the service reliability is differential in the balancing instant, recursive in the number of tasks to be serviced by the DCS, and depends also on the LB policy.

In order to solve the equation in Theorem 1, not only values of $\mathbf{Q}_0, \mathbf{F}_0$, and $\mathbf{C}_0$ for $m_i - 1$ are required, but also other system configurations, such as when only one of the servers is functioning, when more than one group of tasks is in transit to a server, and when no tasks are in transit in the network. Consequently, starting with $\mathbf{Q}_0, \mathbf{F}_0$, and $\mathbf{C}_0$ and (2), we have to construct a system of equations that has to be solved following a particular order. Equations forming such a system are derived in a straightforward manner using (2) and the new initial configurations shown at the right-hand side of (2). Finally, recursions are solved using the initial conditions: $R(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) = 1$ when there are no tasks to process in the DCS and $R(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) = 0$ when all the nodes have failed and at least one task remains unserviced.

Additionally, to solve the recurrence equation in Theorem 1, we first need to calculate its initial condition corresponding to $t_b = 0$, i.e., $R_{\mathbf{K}}(0; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0)$. By exploiting the regenerative theory developed here, we obtain the algebraic recursion presented in Theorem 2.

Theorem 2

Consider an n*-node DCS with initial system configuration* $\mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0$ at $t_b = 0$. *The service reliability satisfies the algebraic recursion*:

$$R_{\mathbf{K}}(0;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)=\sum_{i=1}^{n}\frac{\lambda_{d_i}}{\lambda}R_{\mathbf{K}}\left(0;\mathbf{Q}_0-\delta_{ii},\mathbf{F}_0,\tilde{\mathbf{C}}_0\right)$$
$$+\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\frac{\lambda_{ij}^{F}}{\lambda}R_{\mathbf{K}}\left(0;\mathbf{Q}_0,\mathbf{F}_0^{ji},\mathbf{C}_0\right)$$
$$+\sum_{i=1}^{n}\sum_{j=1}^{g_i}\frac{\tilde{\lambda}_{j,i}}{\lambda}R_{\mathbf{K}}\left(0;\mathbf{Q}_0+f_{ii}l_{ji}\delta_{ii},\mathbf{F}_0,\mathbf{C}_0^{Z_{ji}}\right) \quad (3)$$
$$+\sum_{i=1}^{n}\frac{\lambda_{f_i}}{\lambda}R_{\mathbf{K}}\left(0;\mathbf{Q}_0^{ii},\mathbf{F}_0^{ii},\mathbf{C}_0^{Y_i}\right),$$
$$with\lambda=\sum_{i=1}^{n}\left(\lambda_{d_i}+\lambda_{f_i}+\sum_{j=1}^{g_i}\tilde{\lambda}_{j,i}\right)+\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\lambda_{ij}^{F}.$$

We omit the proof of Theorem 2 since it is similar to that of Theorem 1. (We refer the reader to [14], [15] for a proof in the special case of $n=2$ nodes.)

2.6 Optimal LB Policies for Maximal Reliability

The model for the service reliability given in Theorems 1 and 2 can be used to search for the optimal balancing instant, t_b^*, and the optimal LB policy, $\mathbf{K}^*$, that maximizes the service reliability. Formally, we have

$$(t_b^*,\mathbf{K}^*)\triangleq\arg\max_{(t_b,\mathbf{K})}R_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0) \quad (4)$$

subject to $t_b\geq 0$ and $K_{ij}\in[0,1]$.

We can attempt to solve the optimization problem using the n-node characterization for the reliability given in Theorems 1 and 2; however, computing the reliability using such characterization is computationally expensive for systems with a large number of nodes as the amount of computation grows exponentially with the number of nodes. Namely, since the total number of information states for the DCS is $2^{n(n-1)}$, the complexity in solving Theorem 1 is bounded by $\mathcal{O}\left(2^{n^2}\right)$. In addition, the complexity in solving

Theorem 2 is bounded by $\mathcal{O}(n!)$, because we must consider all the possible orders of arrival of tasks at the underloaded nodes. As an alternative, for DCSs with an arbitrary number of servers, we follow [14], [15] and provide a suboptimal algorithm for LB policies that scales linearly with the number of nodes. The key idea is to decompose an n-node system into several two-node DCSs and exploit our exact characterization for two-node systems.

2.6.1 Algorithm for Devising Optimal LB Policies

Suppose that the jth node is overloaded and recall that $\mathcal{U}_j$ is the collection of candidate receiver nodes as perceived by the jth node. Let K_{jk}^{i} denote the LB gain, calculated at the ith iteration of the algorithm, that is associated with the load transfer from the jth to the kth node. Similarly, let t_b^i denote the LB instant calculated at the ith iteration of the algorithm. Also, let us denote by $U_j'$the set containing all those recipient nodes k, for which K_{jk}^{i} and t_b^i have been already calculated. In addition, let U_j denote the set of recipient nodes ℓ, for which $t_b^{i-1}$and $K_{j\ell}^{i-1}$have been computed. The algorithm for computing the LB policy is described in the following steps:

Initialization

To start the iterations, the algorithm assumes that $U_j = \mathcal{U}_j, U_j'$ is empty and $K_{jk}^0 = 1$ for all $k \in U_j$. Namely, we have assumed that the jth overloaded node can send full load partitions to the recipient nodes.

Repeat

At the ith iteration of the algorithm we select a recipient node, say the kth node, from the collection U_j. The LB gain K_{jk}^i is obtained by considering a two-node system composed of nodes j and k. Thus, upon the execution of LB at t_b, the kth and the jth nodes have loads $\hat{Q}_{k,j}(t_b)$ and

$$Q_j(t_b) \quad - \sum_{\ell \in (U_j \setminus \{k\})} \left\lfloor K_{jl}^{i-1} p_{j\ell} L_j^{ex}(t_b) \right\rfloor - \sum_{l \in U_j'} \left\lfloor K_{j\ell}^{i} p_{j\ell} L_j^{ex}(t_b) \right\rfloor - \left\lfloor K_{jk}^{i} p_{jk} L_j^{ex}(t_b) \right\rfloor,$$

respectively, while $\left\lfloor K_{jk}^i p_{jk} L_j^{ex}(t_b) \right\rfloor$ tasks are assumed to be in transit from the jth to the kth node. After computing the optimal values t_b^i and K_{jk}^i, we update the sets U_j and U_j'as follows: $U_j \leftarrow U_j \setminus \{k\}$ and $U_j' \leftarrow U_j' \cup \{k\}$. These calculations are repeated until LB instants and LB gains of all the nodes in U_j are obtained, i.e., after U_j becomes empty. After this, we set U_j to be equal to U_j and U_j'be empty.

Termination Condition

The ith iteration of the algorithm is repeated until either all the LB gains converge to a certain value or an user-defined maximum number of iterations, N, is executed. The announced LB gains are those obtained after either one of these two termination conditions are met. The announced t_b is the largest balancing instant computed for each pair of nodes at the last iteration of the algorithm.

Algorithm Complexity and Scalability

Suppose that the jth node is overloaded and has to reallocate tasks to η nodes, with $\eta \in [0, n-1]$. Since the LB policy executed by the nodes is distributed, each node has to solve Theorems 1 and 2 individually. For $n = 2$ nodes, the complexity in solving equations in Theorems 1 and 2 is a function of the number of tasks queued at the jth node, i.e., $\mathcal{O}\left(f(m_j)\right)$. Since the jth node decomposed the DCS in η pairs of DCS, the overloaded node has to solve at most η times the optimization problem (4) for $n = 2$. Further, by construction of the algorithm, the jth node has to solve no more than N times such optimization problem. From this, we observe that the complexity of the algorithm is $\mathcal{O}\left(N(n-1)f(m_j)\right)$. In addition, if an exhaustive search in the LB gains is conducted to solve the optimization problem, then $f(m_j)$is bounded by m_j, because no more than $L_j^{ex} = \lfloor m_j - \Lambda_j(\sum_{\ell \in \mathcal{W}_i} \Lambda_\ell)^{-1} \hat{M}_j \rfloor$ LB gains have to be evaluated.

We conclude that the proposed algorithm scales linearly in both the number of nodes in the DCS and the number of tasks queued at the overloaded node. It must be commented that for $n = 2$, we have observed in our simulations and in our prior works [14], [15] that the service reliability exhibits a concave shape as a function of the LB gains. This heuristic can be exploited to search for the optimal LB gains using a bisection algorithm. As the complexity of bisection search algorithms is logarithmic, the complexity in solving the regenerative equations can be bounded by $\log(m_j)$.

SECTION 3 Results

3.1 Distributed Computing Architecture

We have implemented a small-scale DCS testbed to experimentally validate the theoretical achievements shown in this paper. The hardware architecture consists of the computing nodes, the backup nodes, and the communication network. The set of computing nodes comprises heterogeneous processors, such as Pentium II- and Pentium IV-based computers. Some of the computing nodes are dedicated machines, while others are serving as lightly loaded Web, mail, and database servers. Given that the occurrence of a failure at any node is simulated by software, the set of backup nodes is the same set as the set of working nodes. Upon the occurrence of a failure, a computing node is switched from the so-called working state to the failed state. If a node is in the failed state, then it cannot process tasks. The communication network employed in our architecture is the Internet, where the final links connecting the computing nodes are either wired or wireless. On one hand, some communication links connect nodes separated geographically by a large distance; hence, they naturally exhibit a notorious communication delay. On the other hand, for those nodes in the DCS connected by high-speed links, we have introduced some artificial latency by employing traffic shaper applications. Such kind of applications allow us to reduce the actual transfer speed of the network interfaces to slow speeds such as 1,024 to 512 Kbps.

The software architecture of the DCS is divided in three layers: application, task allocation, and communication. Layers are implemented in software using POSIX threads. The application layer executes the application selected to illustrate the distributed processing: matrix multiplication. We have defined the service of a task as the multiplication of one row by a static matrix, which is duplicated in all nodes. To achieve variability in the processing speed of the nodes, the randomness is introduced in the size of each row by independently choosing its arithmetic precision with an exponential distribution. In addition, the application layer updates the QI of each node and determines the failure instants of each node. As part of the latter task, the application layer also switches the state of a node from working to failed. The same layer maintains, at each node, two vectors of $n-1$ components that track the state of the other nodes 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 nodes remain functioning in the system. The task allocation layer executes the LB policy defined for each type of experiment conducted. This layer schedules and triggers the LB instants when task exchange is performed. It also: 1) determines if a node is overloaded with respect to the other nodes in the system; 2) selects which nodes are candidate receiving nodes; and 3) computes the amount of task to transmit to the receiver nodes by solving the recursion (3). In addition, when a node is in the backup state, this layer executes the reallocation of tasks to all the surviving nodes as described in Section 2. Finally, the communication layer of each node handles the transfer of tasks as well as the transfer of QI and FN packets among the nodes. Each node uses the UDP transport protocol to transfer either a QI or an FN packet to the other nodes. The TCP transport protocol is used to transfer tasks between the nodes.

3.2 Maximizing the Reliability of a Two-Node DCS

We have conducted experiments using a dedicated (node 1) and a nondedicated computer (node 2). The nodes are separated by a large geographic distance and communicate through the Internet. The free parameters of the system, namely, the initial workload and the average failure times, were defined to be: $m_1 = 100$ tasks and $m_2 = 50$ tasks, and $\lambda_{f_1}^{-1} = 300$s and $\lambda_{f_2}^{-1} = 200$s. The remaining system parameters were estimated conducting experiments on the two-node DCS: 1) The estimated service rates of each node are $\lambda_{d_1} = 0.8285$ tasks per second (tps) and $\lambda_{d_2} = 1.2453$ tps; 2) The mean arrival times of QI packets and FN packets are $(\lambda_{12}^{Q})^{-1} = (\lambda_{12}^{F})^{-1} = 1.6134$s and $(\lambda_{21}^{Q})^{-1} = (\lambda_{21}^{F})^{-1} = 1.6659$s; and 3) The parameters for the first-order approximation of the average transfer time of tasks are $a_{12} = 0.243$s per task, $b_{12} = 1.613$s, and $a_{21} = 0.336$s per task and b21=1.666s. Fig. 1a shows results of the experiments conducted on our two-node distributed computing testbed for the case of the communication channel linking node 2 to node 1. In the figure, dots

represent measurements of the transfer time of a group of tasks between the nodes, while the straight-line represents the first-order approximation for the average transfer time of tasks. We observe that, for the amount of tasks to be exchanged in the experiments conducted in this paper, a first-order approximation for the average transfer time is valid for the communication channel.

Let us first look at the solution of the initial condition

$$\mathbf{Q}_0 = \begin{pmatrix} m_1 & 0 \\ 0 & m_2 \end{pmatrix}, \mathbf{F}_0 = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix},$$

and $\mathbf{C}_0 = ([0], [0])$at $t_b = 0$. Note that immediately after the execution of the LB policy, the number of tasks remaining queued at the ith node is $r_i = m_i - L_{ij}$, for $i = 1, 2$ and $i \neq j$. Consequently, the initial system configuration is modified as follows:

$$\tilde{\mathbf{Q}}_0 = \begin{pmatrix} r_1 & 0 \\ 0 & r_2 \end{pmatrix}, \tilde{\mathbf{F}}_0 = \mathbf{F}_0 \text{and} \tilde{\mathbf{C}}_0 = ([1L_{21}], [1L_{12}])$$

and the LB policy executed is

$$\mathbf{K} = \begin{pmatrix} 0 & K_{12} \\ K_{21} & 0 \end{pmatrix}.$$

In order to explore all the possible amounts of tasks to exchange among the nodes, we use the formula $L_{jk} = \lfloor K_{jk} m_j \rfloor$. Note that in a two-node DCS, nodes do not have to partition their excess load because there is only one recipient node.

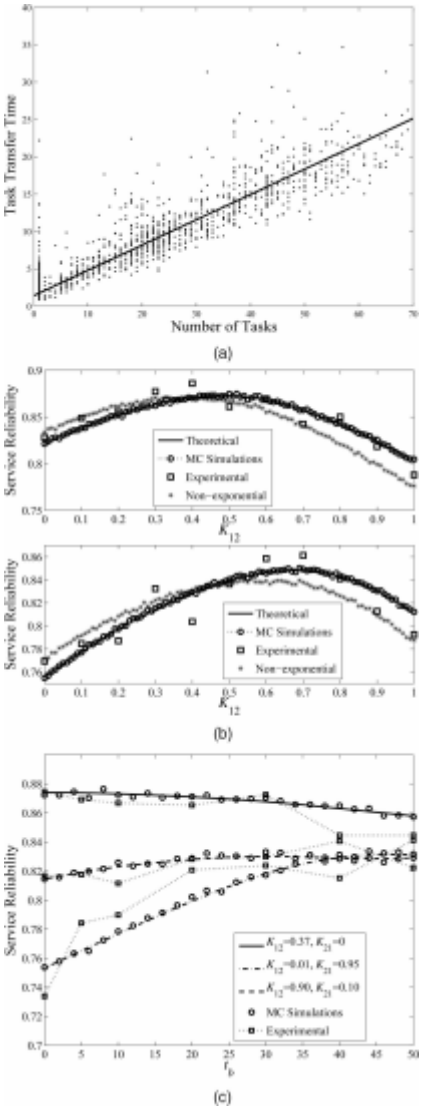

Fig. 1. (a) Dots are realizations of the task transfer time in a two-node DCS. The solid line represents the first-order approximation for the average transfer time. (b) Service reliability as a function of the LB gain of the node, 1 when lb is executed at $t = 0$. In the upper plot $K_{21} = 0.25$ while in the lower plot $K_{21} = 0.9$. (c) Service reliability as a function of the balancing instant for four representative LB gains.

Now we solve the recursion in Theorem 2 to calculate $R_{\mathbf{K}}\left(0; \tilde{\mathbf{Q}}_0, \tilde{\mathbf{F}}_0, \tilde{\mathbf{C}}_0\right)$. In Fig. 1b, the service reliability for $K_{21} = 0.25$ and $K_{21} = 0.9$ are plotted as a function of K_{12}. On one hand, small values for K_{12} imply that node 1 remains unbalanced with respect to node 2 and serves most of its workload. As a consequence, the second node is underutilized because, on average, node 2 serves its entire workload before it fails. Therefore, the time required to serve the workload becomes "large" and the service reliability is "small." On the other hand, when K_{12} approaches 1, the first node transfers most of its initial load to the second node. Hence, almost all the tasks are queued and served at the less reliable node until it fails. Upon failure, the remaining tasks are transferred back to node 1, if it is functioning; thereby, the service reliability is reduced by an excessive queuing of tasks in the communication network. In addition to the theoretical predictions, Fig. 1b shows Monte Carlo (MC) simulations as well as experimental results obtained for the LB policy $\mathbf{K}$. In our simulations, the service reliability is calculated by averaging outcomes (failures or successes) from independent realizations of the policy. The values of reliability plotted in Fig. 1b correspond to centers of 95 percent 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 our theoretical predictions, and remarkably, experiments conducted on the two-node 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 the Fig. 1b.

In order to assess the accuracy of our model, we compare the reliability predicted by our exponential approximation with MC simulations where the distributions of the random times are nonexponential. Using the experimental data collected in our testbed, we have fitted Pareto distributions to the transfer and service times. For comparison, the mean values of the Pareto distributions are equal to the mean values of their corresponding exponential approximations. Via MC simulations, we estimated, with a 95 percent confidence, the service reliability of the DCS when the random times follow Pareto laws. The estimated reliability is plotted in Fig. 1b. It can be noted from the figure that the exponential model for reliability is very accurate and yields a relative approximation error below 4 percent. Further simulations have shown that, as the ratio between the average transfer time and the average service time of the nodes increases, the exponential approximation looses its accuracy in predicting the reliability. Specifically, approximation errors of 120 percent were found when the ratio between the times was five.

Next, we look at the solution of the equations given in Theorem 1. Fig. 1c shows theoretical predictions, MC simulation, and experimental results for the service reliability as a function of the balancing instant, for some representative selections of LB gains. After solving the differential equation in Theorem 1, we obtain a maximal service reliability of 0.874 that is achieved at $t_b^* = 0$ by the following four LB policies:

$$\mathbf{K}_1^* = \begin{pmatrix} 0 & 0.37 \\ 0 & 0 \end{pmatrix}, \mathbf{K}_2^* = \begin{pmatrix} 0 & 0.38 \\ 0 & 0 \end{pmatrix}, \mathbf{K}_3^* = \begin{pmatrix} 0 & 0.39 \\ 0 & 0 \end{pmatrix}, \text{ and } \mathbf{K}_4^* = \begin{pmatrix} 0 & 0.39 \\ 0.02 & 0 \end{pmatrix}.$$

Fig. 1c shows the service reliability as a function of t_b for the optimal policy $\mathbf{K}_1^*$. Note that an improper selection of the LB gains can produce a notorious reduction on the service reliability, as is depicted for the case of choosing

$$\mathbf{K} = \begin{pmatrix} 0 & 0.01 \\ 0.95 & 0 \end{pmatrix}..$$

Note also that, an improper selection of the gains can be compensated by delaying the LB action.

Let us discuss now the effect of the optimal LB policy on the utilization of the computing resources. The optimal policies dictate that 39 percent of the load initially allocated at the first server have to be transferred to the second server, while the latter server must keep all its initial load. Note that, on average, server 2 processes its initial load in 40 s, and note also that, transferring 39 tasks from server 1 to server 2 takes 11 s. Consequently, the optimal task reallocation is perceived by the second server as an instantaneous exchange of load. In addition, note that processing 89 tasks at server 2 takes 71 s, on average, while serving the remaining 61 tasks at server 1 takes 73 s, on average. Therefore, the optimal policy keeps both servers busy for approximately the same amount of time, thereby efficiently using the computing resources of the DCS.

3.3 Maximizing the Reliability of a Multinode DCS

In this section, we maximize the service reliability of a multinode DCS utilizing the algorithm presented in Section 2.6.1. We devise several decentralized LB policies considering different balancing criteria.

The scenario considered in the following examples comprises a five-node DCS, for which a workload of $M = 150$ tasks is provided. We have assumed that the average failure times of the nodes are $\lambda_{f_1}^{-1} = 400\text{s}, \lambda_{f_2}^{-1} = 10\text{s}, \lambda_{f_3}^{-1} = 100\text{s}, \lambda_{f_4}^{-1} = 200\text{s}$, and $\lambda_{f_5}^{-1} = 300\text{s}$. The service rates, estimated using some training sets of tasks on

our DCS testbed, are $\lambda_{d_1} = 0.16823$ tps, $\lambda_{d_2} = 0.49784$ tps, $\lambda_{d_3} = 0.25869$ tps, $\lambda_{d_4} = 0.25361$tps, and $\lambda_{d_5} = 0.18356$ tps. In this scenario, the nodes dedicated only to compute our tasks are the first, the fourth, and the fifth node, while the remaining two are nondedicated nodes. The channel-dependent parameters, also estimated from data collected using our testbed, are listed in Table 1. For brevity, we provide only the minimum and the maximum values for the estimated mean arrival times of both QI and FN packets, namely, $\min\left(\min_{j,k}(\lambda_{jk}^{Q})^{-1}, \min_{j,k}(\lambda_{jk}^{F})^{-1}\right)$=0.343s and $\max\left(\max_{j,k}(\lambda_{jk}^{Q})^{-1}, \max_{j,k}(\lambda_{jk}^{F})^{-1}\right)J = 1.927$s, for $j, k \in \{1, \ldots, 5\}$.

We devise and discuss three LB policies that have the same balancing criterion. The balancing criterion utilized by the policies is based upon the reliability of the nodes. So, we have set the Λ_js parameters in (1) to be $\Lambda_j = \lambda_{f_i}^{-1}$for $j = 1, \ldots, 5$. The three LB policies investigated are: 1) The *Null LB policy*, where all the LB gains employed by the policy are equal to zero; 2) The *Full LB policy*, where all the LB gains are equal to 1; and 3) The *Maximal-Service LB policy*, where the LB gains employed by the policy are computed using the algorithm presented in Section 2.6.1. Note that the Null LB policy determines the service reliability inherently provided by the DCS, i.e., it defines the service reliability when LB is not performed by the nodes in the system. Therefore, the Null LB policy establishes the minimal service reliability that can be demanded by any effective LB policy acting on the DCS.

Table 1 Parameters a_{jk} and b_{jk} of the first-order approximation of the average task-transfer delay for the case of a five-node 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	-

The theoretical predictions obtained for the three LB polices under study, and for different initial task allocations, are listed in Table 2. The service reliability is obtained for each case by means of MC simulations, where the number of tasks to reallocate among the nodes is computed using the algorithm provided in Section 2.6.1. The values in Table 2 correspond to centers of 95 percent confidence intervals, for which the estimated service reliability will not differ from the true value by more than 0.001. In addition, the column labeled as "Exp." presents results obtained after averaging 500 realizations of experiments conducted on our DCS testbed.

The first five rows of Table 2 list results for cases when the system is totally imbalanced. The sixth row presents the case of an initial uniform distribution of tasks. The seventh, eighth, and ninth rows correspond to cases where tasks are initially allocated according to the reliability of the nodes, the processing rate of the nodes, and a combination of the latter two parameters, respectively. Finally, the last row represents a case of an arbitrary task distribution.

We can see from Table 2 that the Maximal-Service LB policy outperforms the other two policies in all the cases considered. In the first five cases listed in Table 2, the optimal tbis equal to zero, while in the remaining cases, the optimal tb is between 2.0 and 2.4 s. Note that such values correspond to cases where all the nodes are, on average, informed about the queue-length of the other nodes in the system. We can also note that the Maximal-Service LB policy effectively increases the inherent service reliability provided by the DCS. Such increment can be attributed mainly to two issues: 1) the Maximal-Service LB policy trades off network queuing times and node idle times by computing appropriate LB gains and 2) the Maximal-Service LB policy effectively exploits the extra balancing action provided by the backup system of a faulty node. To support these statements, we discuss a representative case from Example 2.

Consider the case where all the tasks are queued at the fourth node (fourth row in Table 2). If no LB action is performed, then, on average, at t=200s, the fourth node fails, while, on average, the following events have occurred in the DCS: 1) the second and third nodes have failed; 2) the fourth node has been informed about the failures of the second and third nodes; and 3) the fourth node has served 50 tasks. Upon the failure of the fourth node, its backup system reallocates the remaining 100 tasks to the first and fifth nodes. So, we clearly note that the first and fifth nodes have remained idle for long periods of time, and worst than that, we notice that the second and third nodes were never used to serve any task. On the contrary, if the Full LB policy is employed, then at tb=0, the fourth node decides to transfer 59, 1, 14, and 44 tasks to the first, second, third, and fifth node, respectively, while 32 tasks remain queued at the fourth node. As such, we can deduce from the discussion that the Full LB policy is advantageous over the Null LB policy, as evidenced by the reliability shown in Table 2.

Table 2 Service reliability under different LB policies

Initial load $(m_1, \dots, m_5)$			Max-Service	
	Null Theo.	Full Theo.	Theo.	Exp.
(150,0,0 ,0,0)	0.210	0.508	0.509	0.527
(0,150,0,0,0)	0.552	0.495	0.614	0.595
(0,0,150,0,0)	0.372	0.510	0.601	0.597
(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

The balancing criterion utilized is based on the reliability of the nodes.

Notably, the Maximal-Service LB policy takes an even better decision at the balancing instant by exploiting one extra mechanism. After executing the proposed algorithm, the policy obtains the following LB gains: $K_{41}^{*} = 0.610, K_{42}^{*} = K_{43}^{*} = 1$, and $K_{45}^{*} = 0.886$; this implies that the fourth node has to transfer 35, 1, 14, and 38 tasks to the first, second, third, and fifth node, respectively. Unlike the Full LB policy, a total of 62 tasks remain queued at the fourth node. Note that by sending fewer tasks to the first and fifth nodes, the Maximal-Service LB policy reduces the idle time of these nodes as compared to the Full LB policy. In addition, we can note that, on average: 1) the fourth node is able to process only 50 tasks before it fails and 2) by the average failure time of the fourth node, the first and fifth nodes are still busy serving the tasks reallocated at the balancing instant. Therefore, the task reallocation performed upon the failure of the fourth node does not introduce any idle time in the receiving nodes.

It can be observed from Table 2 that caution must be exercised in selecting the amount of tasks to reallocate at the balancing time; otherwise, we can devise policies that perform worse than taking no LB action! Consider, for instance, the case where all the tasks are queued at the second node. When the Full LB policy is employed, the policy determines that 59, 14, 29, and 44 tasks have to be transferred to the first, third, fourth, and fifth nodes. The four tasks that remain queued at the second node are, on average, served by the node. In addition, the average transfer plus service time of the tasks assigned to the third and fourth nodes is about 60 and 124 s, respectively. We now note that the inappropriate task reallocation performed by the Full LB policy forces the fourth node to remain idle for 76 s before it fails. If we perform a similar kind of analysis for the case of the Null LB policy, we can conclude that, due to the failures of the second and third nodes and the task exchanges performed by their backup systems, the fourth node is kept busy until it fails at $t = 200s$, and its average idle time is only 20 s. Therefore, it can be concluded that the Full LB policy performs worse than the Null LB policy.

Table 3 Service reliability achieved by three LB policies, which have different balancing criteria

Initial load $(m_1, \dots, m_5)$	Service reliability			
	Max-Service	Proc -Speed	Complete	Optimum
(150,0,0,0,0)	0.509	0.511	0.573	0.631
(0,150,0,0,0)	0.614	0.610	0.617	0.617
(0,0,150,0,0)	0.601	0.591	0.601	0.601
(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

For comparison purposes, we list the optimal value obtained for each case.

In light of the previous discussions, we can now comprehend the counterintuitive behavior shown in Table 2. It can be noted that, for cases where all the load is queued at a single node and no LB action is taken, the service reliability is better when tasks are initially allocated at the less reliable nodes. This situation is justified because, by initially allocating the workload at less reliable nodes, we can exploit both the computing power of the unreliable servers and the task reallocation action executed by the backup system of the faulty nodes.

We study now the effect of the selection of various balancing criteria on the service reliability. We have considered three LB policies, each one of them having a different balancing criterion but sharing the same algorithm to compute the LB gains. The Maximal-Service LB policy balances the DCS according to the reliability of the nodes. The *Processing Speed LB policy* balances the DCS based upon the processing rate of the nodes, i.e., $\Lambda_j = \lambda_d$ in (1). Finally, the *Complete LB policy* uses a balancing criterion that combines both processing and failure rates, specifically, the Complete LB policy defines $\Lambda_j = \lambda_{d_j}(1 - \lambda_{f_j}(\sum_{k=1}^{n} \lambda_{f_k})^{-1})$. Additionally, we have conducted MC-based exhaustive search, over the LB gains, in order to estimate the optimal service reliability for each case considered. The results of our evaluations are listed in Table 3.

Note that the fastest servers in the example are also the less reliable ones. Consequently, the balancing 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 3 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 node, the Processing Speed LB policy dictates that 54 tasks have to be transferred to the second node. However, the LB gain computed by our algorithm reduces such amount to only 11 tasks. From Table 3, we observe that the Complete LB policy outperforms in almost all the cases the

other two policies. This is because such a policy trades off reliability and computing speed in both the imbalance detection process and the excess workload partitioning. Finally, it can be seen from Table 3 that the service reliability achieved by the policies is within 70 percent of the optimal service reliability for each case. In fact, the optimum is achieved in some cases.

SECTION 4 Conclusions

We have undertaken a novel approach to analyze the stochastic dynamics and the service reliability of DCSs in the presence of communication and node uncertainty. We have rigorously modeled the service reliability of a DCS, i.e., the probability of successfully serving a collection of tasks queued at the nodes before all of them fail permanently. Our model takes into account the heterogeneity in the computing resources, the stochastic communication and transfer delays in the network, the uncertainty associated with the number of functional nodes in the DCS, and an arbitrary LB policy executed by the nodes. We have introduced in our analysis three fundamental stochastic quantities, namely the system queue and the system function matrices as well as the network state vector. These quantities track the underlying point processes associated with the dynamics of the DCS. At any given time, these matrices store information about task distribution among the nodes, the functional or dysfunctional state of the nodes, and the number of tasks queued in the communication network. A novel regeneration argument has been established yielding an analytic characterization for the service reliability. Our mathematical framework can be easily modified to calculate other performance metrics, such as computing speedup, statistics of queue length of servers, and average sojourn time of workloads.

By using this analytical model for reliability, we have devised optimal dynamic LB strategies for maximizing the service reliability of a DCS. We have presented a simple, yet efficient and scalable algorithm for devising these optimal dynamic LB strategies. The policies devised using our algorithm dictate when to execute the LB action and how to reallocate the tasks among the nodes. We have evaluated the performance of the optimal LB policies and noticed that the service reliability can be improved up to 65 percent as compared to the reliability provided by a DCS, and up to 22 percent as compared to policies that consider nodes' reliability but disregard the communication costs over the network. Moreover, our algorithm to compute the LB strategies achieves a service reliability within 70 percent of the optimal service reliability, and in cases achieves the optimal value.

Our theory enables us to understand the effectiveness of task reallocation in a delay-infested DCS while offering an algorithm for generating task reallocation policies that maximize the service reliability. The interplay between the task transfer time and the idle time of the nodes has been discussed, and we have noted that the service reliability can be improved if the idle times of the nodes are reduced as much as possible. In addition, we have discussed the advantages of delaying the balancing action until the nodes have collected information about both the queue length and the functioning state of the nodes.

In general, we have found that the experiments confirm our theoretical predictions as well as our MC simulations. Through experimentation, we have also observed that the computational overhead introduced by our algorithm, which is mainly due to the calculations associated to the regenerative equations, is negligible as compared to the time to serve the tasks.

Future work will consider relaxing the exponential assumption on the random task transfer and task execution times. To this end, we will undertake an age-dependent regeneration-based approach whereby auxiliary “age-variables” are introduced in the analysis. Another extension we are currently considering is to allow each node have an arbitrary number of functionality states instead of a binary (on-off) functionality state as presented here. This can be implemented, for example, by assigning a range of possible processing speeds for each node, where upon the occurrence of a “failure event,” only one of these possible states is selected.

ACKNOWLEDGEMENT

This work was supported by the Defense Threat Reduction Agency (Combating WMD Basic Research Program) and in part by the US National Science Foundation (NSF) (Award ANI-0312611). The author (Sagar Dhakal) is an ASEE Postdoctoral Fellow in the Acoustic and Signal Processing Branch at NRL. This work was conducted before he joined NRL and does not represent any view from NRL.

The gist of the proof of Theorem 1 can be found in [14] for the special case of two nodes. Here, we present a generalized version of such proof considering a DCS with n nodes.

For clarity, we first introduce some useful definitions and then present Lemmas 1–6, which will be used in the proof of Theorem 1. Recall that the regeneration time is defined as

$$\tau \triangleq \min\left(\min_k(W_{k1}), \min_{j\neq k}(X_{jk}^{Q}), \min_k(Y_k), \min_{j\neq k}(X_{jk}^{F}), \min_{k,i}(Z_{ki})\right).$$

Note that in light of Assumptions A1, A2, and Convention C1, it is straightforward to see that τ is an exponentially distributed random variable. For the DCS emerging at the regeneration time τ, let the random times (all measured from τ) $W'_{ki}, Y'_k, X_{jk}^{Q'}, X_{jk}^{F'}$, and Z'_{ik}, respectively, be the service time for the ith task at the kth node, the failure time of the kth node, the arrival time of the QI packet sent from the jth node to the kth node, the arrival time of the FN packet sent from the jth node to the kth node, and the arrival time of the ith group of tasks sent to the kth node. In addition, on $\{\tau \leq t_b\}$, we define $T'_{\mathbf{K}}(t_b; \mathbf{Q}'_0, \mathbf{F}'_0, \mathbf{C}'_0)$ as the time taken by the new DCS emerging at τ to serve all the tasks in the system if the LB policy $\mathbf{K}$ is executed by all functioning nodes at time t_b provided that the system condition at $t = \tau$ is specified by $\mathbf{Q}'_0$, $\mathbf{F}'_0$, and $\mathbf{C}'_0$. To prove that the DCS is regenerated upon the occurrence of $\{\tau = s\}$, it suffices to show that the conditional distributions of $W'_{ki}, X_{jk}^{Q'}, Y'_k, X_{jk}^{F'}$, and Z'_{ki} given that the event $\{\tau = s\}$ has occurred, satisfy assumptions Al and A2.

Lemma 1

For $s \leq t_b$, $\mathrm{P}\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty | \tau = s, \tau = W_{i1}\} = \mathrm{P}\{T_{\mathbf{K}}(t_b - s; \mathbf{Q}_0 - \delta_{ii}, \mathbf{F}_0, \mathbf{C}_0) < \infty\}$**.**

Proof

Note that the regeneration event $\{\tau = s, \tau = W_{i1}\}$ is precisely service to the first task at the ith node before any other activity takes place in the DCS. Upon the occurrence of the event $\{\tau = s, \tau = W_{i1}\}$, the system function state and the network state remain the same, i.e., $\mathbf{F}'_0 = \mathbf{F}_0$ and $\mathbf{C}'_0 = \mathbf{C}_0$, while $m_i - 1$ tasks are now queued at the ith node and m_j remain queued at the jth node, $j \neq i$. In matrix notation, i.e., $\mathbf{Q}'_0 = \mathbf{Q}_0 - \delta_{ii}$. Therefore, by construction,

$$\mathrm{P}\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty | \tau = s, \tau = W_{i1}\} = \mathrm{P}\{\tau + T'_{\mathbf{K}}(t_b; \mathbf{Q}_0 - \delta_{it}, \mathbf{F}_0, \mathbf{C}_0) < \infty | \tau = s, \tau = W_{i1}\}.$$

The proof is complete once we establish that

$$\begin{aligned}\mathrm{P}\{\quad & T'_{\mathbf{K}}(t_b;\mathbf{Q}_0-\delta_{ii},\mathbf{F}_0,\mathbf{C}_0)<\infty|\tau=s,\tau=W_{x1}\}\\ &=\mathrm{P}\{T_{\mathbf{K}}(t_b-s;\mathbf{Q}_0-\delta_{ii},\mathbf{F}_0,\mathbf{C}_0)<\infty\}.\end{aligned}$$

Next, by construction, $W'_{k1}=W_{k1}-\tau, X_{ik}^{Q'}=X_{jk}-\tau, Y'_j=Y_j-\tau, X_{jk}^{F'}=X_{jk}^{F}-\tau$, and $Z'_{\ell k}=Z_{\ell k-\tau}$ for $k\neq i$ and $j\neq k$. Moreover, it is shown below that the conditional distribution of W'_{k1} is

$$\mathrm{P}\{W'_{k1}\leq t|\tau=s,\tau=W_{i1}\}=\left(1-\exp\left(-\lambda_{d_k}t\right)\right)u(t), \quad (5)$$

where $u(\cdot)$ is the unit step function. Similarly, we get

$$\begin{aligned}\mathrm{P}\quad & \left\{X_{jk}^{Q'}\leq t\middle|\tau=s,\tau=W_{i1}\right\}=\left(1-\exp\left(-\lambda_{jk}^{Q}t\right)\right)u(t),\\ & \mathrm{P}\{Y'_k\leq t|\tau=s,\tau=W_{i1}\}=\left(1-\exp\left(-\lambda_{f_k}t\right)\right)u(t),\\ & \mathrm{P}\left\{X_{jk}^{F'}\leq t\middle|\tau=s,\tau=W_{i1}\right\}=\left(1-\exp\left(-\lambda_{jk}^{F}t\right)\right)u(t), \text{ and}\\ & \mathrm{P}\left\{Z'_{jk}\leq t\middle|\tau=s,\tau=W_{i1}\right\}=\left(1-\exp\left(-\tilde{\lambda}_{j,k}t\right)\right)u(t).\end{aligned}$$

Therefore, conditional upon the occurrence of $\{\tau=s,\tau=W_{i1}\}$, all random times of the newly emerging DCS satisfy Assumption A1.

The conditional independence of W'_{j1}, with $j\neq i$, and Y'_k is proved below in this Appendix. Similarly, it can also be shown that conditional upon the occurrence of $\{\tau=s,\tau=W_{i1}\}$, the random times $W'_{kj}, X_{jk}^{Q'}, X_{jk}^{F'}$, and Z'_{jk} are mutually independent. Therefore, upon the occurrence of $\{\tau=s,\tau=W_{i1}\}$, all random times of the emerging DCS satisfy Assumption A2.

In summary, we have shown that conditional on the occurrence of $\{\tau=s,\tau=W_{i1}\}$, the random times characterizing the DCS at time s satisfy Assumptions Al and A2. Therefore, by shifting the time origin from $t=0$ to $t=s$, we can think of the emergent DCS as the original system but with m_i-1 tasks in the queue of the ith node, while other system initial conditions remain the same. In addition, due to the shift of origin, the LB instant is now at t_b-s units of time from the new origin. Hence, we conclude that

$$\begin{aligned}\mathrm{P}\{\quad & T'_{\mathbf{K}}(t_b;\mathbf{Q}_0-\delta_{ii},\mathbf{F}_0,\mathbf{C}_0)<\infty|\tau=s,\tau=W_{i1}\}\\ &=\mathrm{P}\{T_{\mathbf{K}}(t_b-s;\mathbf{Q}_0-\delta_{ii},\mathbf{F}_0,\mathbf{C}_0)<\infty\},\end{aligned}$$

which completes the proof of Lemma 1. □

Lemmas 2–5 are presented without proof as they follow similar principles as those of Lemma 1.

Lemma 2

For $s\leq t_b$, $\mathrm{P}\left\{T_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)<\infty\middle|\tau=s,\tau=X_{ij}^{Q}\right\}=\mathrm{P}\left\{T_{\mathbf{K}}\left(t_b-s;\mathbf{Q}_0+\delta_{ji},\mathbf{F}_0,\mathbf{C}_0\right)<\infty\right\}$.

Lemma 3

For $s\leq t_b$, $\mathrm{P}\{T_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)<\infty|\tau=s,\tau=Y_i\}=\mathrm{P}\left\{T_{\mathbf{K}}\left(t_b-s;\mathbf{Q}_0^{ii},\mathbf{F}_0^{ii},\mathbf{C}_0^{Y_i}\right)<\infty\right\}$.

Lemma 4

For $s \leq t_b$, $\mathrm{P}\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty | \tau = s, \tau = X_{ij}^F\} = \mathrm{P}\{T_{\mathbf{K}}(t_b - s; \mathbf{Q}_0^{ji}, \mathbf{F}_0^{ji}, \mathbf{C}_0) < \infty\}$.

Lemma 5

For $s \leq t_b$, $\mathrm{P}\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty | \tau = s, \tau = Z_{ji}\} = \mathrm{P}\left\{T_{\mathbf{K}}\left(t_b - s; \mathbf{Q}_0 + f_{ii} l_{ji} \delta_{ii}, \mathbf{F}_0, \mathbf{C}_0^{Z_{ji}}\right) < \infty\right\}$.

Lemma 6

$\mathrm{P}\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty | \tau > t_b\} = \mathrm{P}\{T_{\mathbf{K}}(0; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty\}$.

Proof

The occurrence of the event $\{\tau > t_b\}$ implies that the system condition of the DCS at time t_b is exactly the same as the initial system condition of the original system. Therefore, for $\{\tau > t_b\}$, let $T''_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0)$ be the time taken by the new DCS emerging at tb to serve all tasks if the LB policy $\mathbf{K}$ is executed by all functioning nodes at time t_b, and provided that the system condition at $t = t_b$ is specified by $\mathbf{Q}_0, \mathbf{F}_0$, and $\mathbf{C}_0$.

Therefore, by construction,

$\mathrm{P}\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty | \tau > t_b\} = \mathrm{P}\{t_b + T''_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty | \tau > t_b\}$.

Let the random times characterizing the DCS emerging at t_b be $W'_{kj}, X_{jk}^{Q'}, X_{jk}^{F'}$, and Z'_{jk}, all measured from t_b. We have that $W''_{i1} = W_{i1} - t_b, X_{jk}^{Q''} = X_{jk}^Q - t_b, Y''_k = T_k - t_b, X_{jk}^{F''} = X_{jk}^F - t_b$, and $Z''_{ik} = Z_{ik} - t_b$. Based on Assumptions Al and A2, it is straightforward to show that $\mathrm{P}\{W''_{hi} \leq t | \tau > t_b\} = (1 - \exp(-\lambda_{d_k} t))u(t)$ and $\mathrm{P}\{W''_{ki} \leq t_1, Y''_k \leq t_2 | \tau > t_b\} = \mathrm{P}\{W''_{ki} \leq t_1 | \tau > t_b\}\mathrm{P}\{Y''_k \leq t_2 | \tau > t_b\}$. Similarly, conditional on the occurrence of $\{\tau > t_b\}$, the conditional distributions of $X_{jk}^{Q''}, Y''_k, X_{jk}^{F''}$, and Z''_{ki} can be shown to satisfy Al and A2. Consequently, nothing has changed in the initial condition as well as the statistics of the random times characterizing the DCS while t_b units of time have elapsed. Therefore, we can shift the origin by t_b units of time, which makes the LB instant at $t = 0$ for the new DCS. So, $\mathrm{P}\{T''_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty | \tau > t_b\} = \mathrm{P}\{T_{\mathbf{K}}(0; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty\}$. □

Proof of Theorem 1

First we observe that from Assumptions Al and A2, it is straightforward to show that $f_\tau(t) = \lambda \exp(-\lambda t)u(t)$, where $\lambda = \sum_{i=1}^n (\lambda_{d_i} + \lambda_{f_i} + \sum_{j=1}^{g_i} \tilde{\lambda}_{j,i} + \sum_{j \neq k} (\lambda_{jk}^Q + \lambda_{jk}^F))$. Next, we condition the service reliability on the regeneration time to obtain

$$\begin{aligned} R_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) &= \mathrm{P}\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty\} \\ &= \int_0^{t_b} \mathrm{P}\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty | \tau = s\} f_\tau(s) ds \quad (6) \\ &+ \int_{t_b}^{\infty} \mathrm{P}\{T_{\mathbf{K}}(t_b; \mathbf{Q}_0, \mathbf{F}_0, \mathbf{C}_0) < \infty | \tau = s\} f_\tau(s) ds\,. \end{aligned}$$

Moreover, we can further condition the first integrand at the right side of (6) on all the possible, disjoint regeneration events occurring at the time $\tau = s$ as

$$
\begin{aligned}
\mathrm{P}\{ & T_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)<\infty|\tau=s\}=\textstyle\sum_{i=1}^{n}\ \mathrm{P}\{\tau=W_{i1}|\tau=s\}\\
\times\ & \mathrm{P}\{T_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)<\infty|\tau=s,\tau=W_{i1}\}\\
+\ & \textstyle\sum_{i=1}^{n}\ \sum_{j=1,j\neq i}^{n}\ \mathrm{P}\{T_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)<\infty|\tau=s,\tau=X_{ij}^{Q}\}\\
& \mathrm{P}\{\tau=X_{ij}|\tau=s\}+\textstyle\sum_{i=1}^{n}\ \mathrm{P}\{T_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)<\infty|\tau=s,\\
& \tau=Y_i\}\mathrm{P}\{\tau=Y_i|\tau=s\}+\textstyle\sum_{i=1}^{n}\ \sum_{j=1,j\neq i}^{n}\ \mathrm{P}\{T_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)\cdot \quad (7)\\
& <\infty|\tau=s,\tau=X_{ij}^{F}\}\times\mathrm{P}\{\tau=X_{ij}^{F}|\tau=s\}\\
+\ & \textstyle\sum_{i=1}^{n}\ \sum_{j=1}^{g_x}\ \mathrm{P}\{\tau=Z_{ji}|\tau=s\}\times\mathrm{P}\{T_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)\\
& <\infty|\tau=s,\tau=Z_{ji}\}.
\end{aligned}
$$

In addition, note that $\int_{t_b}^{\infty}\ \mathrm{P}\{T_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)<\infty|\tau=s\}f_\tau(s)ds=\mathrm{P}\{T_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)<\infty|\tau>t_b\}\mathrm{P}\{\tau>t_b\}$. We now apply Lemma 6 to the later result and Lemmas 1–5 to **(7)**, and substitute those results in **(6)** to obtain:

$$
\begin{aligned}
R_{\mathbf{K}}\ & (t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)=\textstyle\int_0^{t_b}(\sum_{i=1}^{n}\ \mathrm{P}\{\tau=W_{i1}|\tau=s\}\\
& \times R_{\mathbf{K}}(t_b-s;\mathbf{Q}_0-\delta_{ii},\mathbf{F}_0,\mathbf{C}_0)+\textstyle\sum_{i=1}^{n}\ \sum_{j=1,j\neq i}^{n}\ \mathrm{P}\{\tau=X_{ij}^{Q}|\tau=s\}\\
& \times R_{\mathbf{K}}(t_b-s;\mathbf{Q}_0+\delta_{ji},\mathbf{F}_0,\mathbf{C}_0)+\textstyle\sum_{i=1}^{n}\ \mathrm{P}\{\tau=Y_i|\tau=s\}\\
& \times R_{\mathbf{K}}(t_b-s;\mathbf{Q}_0^{ii},\mathbf{F}_0^{ii},\mathbf{C}_0^{Y_i})+\textstyle\sum_{i=1}^{n}\ \sum_{j=1,j\neq i}^{n}\ \mathrm{P}\{\tau=X_{ij}^{F}|\tau=s\}\quad .\\
& \times R_{\mathbf{K}}(t_b-s;\mathbf{Q}_0^{ji},\mathbf{F}_0^{ji},\mathbf{C}_0)+\textstyle\sum_{i=1}^{n}\ \sum_{j=1}^{g_i}\ \mathrm{P}\{\tau=Z_{ji}|\tau=s\}\\
& \times R_{\mathbf{K}}(t_b-s;\mathbf{Q}_0+f_{ii}l_{ji}\delta_{ii},\mathbf{F}_0,\mathbf{C}_0^{Z_{ji}}))f_\tau(s)ds\\
& +\mathrm{P}\{\tau>t_b\}R_{\mathbf{K}}(0;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0).
\end{aligned}
$$

Using basic concepts from probability theory, we can show that $\mathrm{P}\{\tau=W_{i1}|\tau=s\}=\lambda_{d_i}\lambda^{-1}$,$\mathrm{P}\{\tau=X_{jk}^{Q}|\tau=s\}=\lambda_{jk}^{Q}\lambda^{-1}$, $\mathrm{P}\{\tau=Y_k|\tau=s\}=\lambda_{f_k}\lambda^{-1}$, $\mathrm{P}\{\tau=X_{jk}^{F}|\tau=s\}=\lambda_{jk}^{F}\lambda^{-1}$,$\mathrm{P}\{\tau=Z_{ji}|\tau=s\}=\tilde{\lambda}_{ji}\lambda^{-1}$, Therefore, the last equation becomes

$$
\begin{aligned}
& R_{\mathbf{K}}(t_b;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0)=\textstyle\int_0^{t_b}\ \exp(-\lambda s)\\
& (\textstyle\sum_{i=1}^{n}\ \lambda_{d_i}R_{\mathbf{K}}(t_b-s,\mathbf{Q}_0-\delta_{ii},\mathbf{F}_0,\mathbf{C}_0)\\
& +\textstyle\sum_{i=1}^{n}\ \sum_{j=1,j\neq i}^{n}\ \lambda_{ij}^{Q}R_{\mathbf{K}}(t_b-s;\mathbf{Q}_0+\delta_{ji},\mathbf{F}_0,\mathbf{C}_0)\\
& +\textstyle\sum_{i=1}^{n}\ \lambda_{f_i}R_{\mathbf{K}}(t_b-s;\mathbf{Q}_0^{ii},\mathbf{F}_0^{ii},\mathbf{C}_0^{Y_i}) \qquad (8)\\
& +\textstyle\sum_{i=1}^{n}\ \sum_{j=1,j\neq i}^{n}\ \lambda_{ij}^{F}R_{\mathbf{K}}(t_b-s;\mathbf{Q}_0^{ji},\mathbf{F}_0^{ji},\mathbf{C}_0)\\
& +\textstyle\sum_{i=1}^{n}\ \sum_{j=1}^{g_i}\ \tilde{\lambda}_{ji}R_{\mathbf{K}}(t_b-s;\mathbf{Q}_0+f_{ii}l_{ji}\delta_{ii},\mathbf{F}_0,\mathbf{C}_0^{Z_{ji}}))\mathrm{d}s\\
& +\mathrm{P}\{\tau>t_b\}R_{\mathbf{K}}(0;\mathbf{Q}_0,\mathbf{F}_0,\mathbf{C}_0).
\end{aligned}
$$

Finally, by differentiating **(8)** with respect to t_b and rearranging terms we obtain **(2)**. □

Proof of Equation (5)

Let us look at the conditional distribution of W'_{j1}, with $j \neq i$:

$$\begin{aligned}\mathrm{P}\{W'_{j1} \leq\ & t|\tau = s, \tau = W_{i1}\} = \mathrm{P}\{W_{j1} - \tau \leq t|\tau = s, \tau = W_{i1}\} \\ & = \mathrm{P}\{W_{j1} \leq t + s|\tau = s, \tau = W_{i1}\}.\end{aligned}$$

Note that the event $\{\tau = s, \tau = W_{i1}\}$ is equivalent to

$$\begin{aligned}\{W_{i1} =\ & s, W_{11} > s, \dots, W_{(i-1)1} > s, W_{(i+1)1} > s, \dots W_{n1} \\ & > s, Y_1 > s, \dots, Y_n > s, X^Q_{12} > s, \dots, X^Q_{n(n-1)} \\ & > s, X^F_{12} > s, \dots, X^F_{n(n-1)} > s, Z_{11} > s, \dots, Z_{ng_n} > s\}.\end{aligned}$$

Therefore, the latter equation becomes

$$\begin{aligned}\mathrm{P}\{W'_{j1}\ & \leq t|\tau = s, \tau = W_{i1}\} = \mathrm{P}\{W_{j1} \leq t + s|W_{i1} = s, \\ & W_{11} > s, \dots, W_{(i-1)1} > s, W_{(i+1)1} > s, \dots, W_{n1} > s, \\ & Y_1 > s, \dots, Y_n > s, X^Q_{12} > s, \dots, X^Q_{n(n-1)} > s, \\ & X^F_{12} > s, \dots, X^F_{n(n-1)} > s, Z_{11} > s, \dots, Z_{ng_n} > s\}.\end{aligned}$$

Exploiting the independence (Assumption A2), we obtain $\mathrm{P}\{W'_{j1} \leq t|\tau = s, \tau = W_{i1}\} = \mathrm{P}\{W_{j1} \leq t + s|W_{j1} > s\} = (1 - \exp(-\lambda_{d_j} t))u(t)$. □

Proof of the conditional independence of W'_{j1} and Y'_k. Recall that $W'_{j1} = W_{j1} - \tau$ and $Y'_k = Y_k - \tau$. Therefore, for any real number t_1 and t_2, we have

$$\begin{aligned}\mathrm{P}\{W'_{j1}\ & \leq t_1, Y'_k \leq t_2|\tau = s, \tau = W_{i1}\} \\ = \mathrm{P}\ & \{W_{j1} \leq t_1 + s, Y_k \leq t_2 + s|\tau = s, \tau = W_{i1}\} \\ = \mathrm{P}\ & \{W_{j1} \leq t_1 + s, Y_1 \leq t_2 + s|W_{i1} = s, W_{11} > s, \dots, \\ & W_{(i-1)1} > s, W_{(i+1)1} > s, \dots, W_{n1} > s, Y_1 > s, \dots, \\ & Y_n > s, X^Q_{12} > s, \dots, X^Q_{n(n-1)} > s, X^F_{12} > s, \dots, \\ & X^F_{n(n-1)} > s, Z_{11} > s, \dots, Z_{ng_n} > s\},\end{aligned}$$

since the events conditioning the probability are equivalent. Next, by exploiting Assumption A2, we have

$$
\begin{aligned}
\mathrm{P}\{ \quad & W'_{j1} \le t_1, Y'_k \le t_2 | \tau = s, \tau = W_{i1}\} \\
&= \mathrm{P}\{W_{j1} \le t_1 + s, Y_k \le t_2 + s | \tau = s, \tau = W_{i1}\} \\
&= \mathrm{P}\{W_{j1} \le t_1 + s, Y_k \le t_2 + s | W_{j1} > s, Y_k > s\} \\
&= \frac{\mathrm{P}\{W_{j1} \le t_1 + s, Y_k \le t_2 + s, W_{j1} > s, Y_k > s\}}{\mathrm{P}\{W_{j1} > s, Y_k > s\}} \\
&= \frac{\mathrm{P}\{s < W_{j1} \le t_1 + s\}}{\mathrm{P}\{W_{j1} > s\}} \frac{\mathrm{P}\{s < Y_k \le t_2 + s\}}{\mathrm{P}\{Y_k > s\}}.
\end{aligned}
$$

Therefore, we get $\mathrm{P}\{W'_{j1} \le t_1, Y'_k \le t_2 | \tau = s, \tau = W_{i1}\} = \mathrm{P}\{W'_{j1} \le t_1 | W_{j1} > s\}\mathrm{P}\{Y'_k \le t_2 | Y_k > s\}$, which concludes the proof by noting that $\mathrm{P}\{W'_{j1} \le t_1 | \tau = s, \tau = W_{i1}\} = \mathrm{P}\{W'_{j1} \le t_1 | W_{j1} > s\}$. □

References

1. R. Shah, B. Veeravalli, M. Misra, "On the Design of Adaptive and Decentralized Load Balancing Algorithms with Load Estimation for Computational Grid Environments", *IEEE Trans. Parallel and Distributed Systems*, vol. 18, no. 12, pp. 1675-1686, Dec. 2007.

2. L. Tassiulas, A. Ephremides, "Stability Properties of Constrained Queuing Systems and Scheduling Policies for Maximum Throughput in Multihop Radio Networks", *IEEE Trans. Automatic Control*, vol. 37, no. 12, pp. 1936-1948, Dec. 1992.

3. M. Neely, E. Modiano, C. Rohrs, "Dynamic Power Allocation and Routing for Time Varying Wireless Networks", *Proc. IEEE INFOCOM*, 2003.

4. G. Koole, P. Sparaggis, D. Towsley, "Minimizing Response Times and Queue Lengths in Systems of Parallel Queues", *J. Applied Probability*, vol. 36, pp. 1185-1193, 1999.

5. L. Golubchik, J. Lui, R. Muntz, "Chained Declustering: Load Balancing and Robustness to Skew and Failures", *Proc. Workshop Research Issues on Data Eng.*, pp. 88-95, 1992.

6. A. Brandt, M. Brandt, "On a Two-Queue Priority System with Impatience and Its Application to a Call Center", *Methodology and Computing in Applied Probability*, vol. 1, pp. 191-210, 1999.

7. M. Hayat, S. Dhakal, C. Abdallah, J. Birdwell, J. Chiasson, "Advances in Time Delay Systems" in Dynamic Time Delay Models for Load Balancing. Part II. Stochastic Analysis of the Effect of Delay Uncertainty, Springer-Verlag, pp. 355-368, 2004.

8. S. Dhakal, M. Hayat, J. Pezoa, C. Yang, D. Bader, "Dynamic Load Balancing in Distributed Systems in the Presence of Delays: A Regeneration-Theory Approach", *IEEE Trans. Parallel and Distributed Systems*, vol. 18, no. 4, pp. 485-497, Apr. 2007.

9. S. Dhakal, M. Hayat, J. Pezoa, C. Abdallah, J. Birdwell, J. Chiasson, "Load Balancing in the Presence of Random Node Failure and Recovery", *Proc. IEEE Int'l Parallel and Distributed Processing Symp. (IPDPS)*, 2006.

10. Y.-S. Dai, G. Levitin, "Optimal Resource Allocation for Maximizing Performance and Reliability in Tree-Structured Grid Services", *IEEE Trans. Reliability*, vol. 56, no. 3, pp. 444-453, Sept. 2007.

11. Y.-S. Dai, G. Levitin, K. Trivedi, "Performance and Reliability of Tree-Structured Grid Services Considering Data Dependence and Failure Correlation", *IEEE Trans. Computers*, vol. 56, no. 7, pp. 925-936, July 2007.

12. G. Attiya, Y. Hamam, "Reliability Oriented Task Allocation in Heterogeneous Distributed Computing Systems", *Proc. Ninth Int'l Symp. Computers and Comm.*, pp. 68-73, 2004.

13. C.-I. Chen, "Task Allocation and Reallocation for Fault Tolerance in Multicomputer Systems", *Trans. Aerospace and Electronic Systems*, vol. 30, pp. 1094-1104, 1994.

14. S. Dhakal, *Load Balancing in Communication Constrained Distributed Systems: A Probabilistic Approach*, 2006.
15. J. Pezoa, S. Dhakal, M. Hayat, "Decentralized Load Balancing for Improving Reliability in Heterogeneous Distributed Systems", *Proc. Int'l Conf. Parallel Processing (ICPP)*, 2009.
16. V. Shestak, J. Smith, A. Maciejewski, H. Siegel, "Stochastic Robustness Metric and Its Use for Static Resource Allocations", *J. Parallel and Distributed Computing*, vol. 68, pp. 1157-1173, 2008.
17. M. Trehel, C. Balayer, A. Alloui, "Modeling Load Balancing Inside Groups Using Queuing Theory", *Proc. 10th Int'l Conf. Parallel and Distributed Computing Systems*, 1997.
18. C. Hui, S. Chanson, "Hydrodynamic Load Balancing", *IEEE Trans. Parallel and Distributed Systems*, vol. 10, no. 11, pp. 1118-1137, Nov. 1999.
19. Z. Lan, V. Taylor, G. Bryan, "Dynamic Load Balancing for Adaptive Mesh Refinement Application", *Proc. Int'l Conf. Parallel Processing (ICPP)*, 2001.
20. S. Dhakal, B. Paskaleva, M. Hayat, E. Schamiloglu, C. Abdallah, "Dynamical Discrete-Time Load Balancing in Distributed Systems in the Presence of Time Delays", *Proc. IEEE Conf. Decision and Control (CDC)*, 2003.
21. H. Lee, S. Chin, J. Lee, D. Lee, K. Chung, S. Jung, H. Yu, "A Resource Manager for Optimal Resource Selection and Fault Tolerance Service in Grids", *Proc. IEEE Int'l Symp. Cluster Computing and the Grid (ISCCG)*, 2004.
22. M. Litzkow, M. Livny, M. Mutka, "Condor—A Hunter of Idle Workstations", *Proc. Int'l Conf. Distrbuted Computing Systems (ICDCS)*, pp. 104-111, 1988.
23. R. Sheahan, L. Lipsky, P. Fiorini, "The Effect of Different Failure Recovery Procedures on the Distribution of Task Completion Times", *Proc. Workshop Dependable Parallel Distributed and Network-Centric Systems (DPDNS)*, 2005.
24. J. Palmer, I. Mitrani, "Empirical and Analytical Evaluation of Systems with Multiple Unreliable Servers", *Proc. Int'l Conf. Dependable Systems and Networks*, pp. 517-525, 2006.
25. S. Shatz, J.-P. Wang, "Models and Algorithms for Reliability-Oriented Task-Allocation in Redundant Distributed-Computer Systems", *IEEE Trans. Reliability*, vol. 38, no. 1, pp. 16-27, Apr. 1989.
26. V. Ravi, B. Murty, J. Reddy, "Nonequilibrium Simulated-Annealing Algorithm Applied to Reliability Optimization of Complex Systems", *IEEE Trans. Reliability*, vol. 46, no. 2, pp. 233-239, June 1997.
27. S. Srinivasan, N. Jha, "Safety and Reliability Driven Task Allocation in Distributed Systems", *IEEE Trans. Parallel and Distributed Systems*, vol. 10, no. 3, pp. 238-251, Mar. 1999.
28. D. Vidyarthi, A. Tripathi, "Maximizing Reliability of a Distributed Computing System with Task Allocation Using Simple Genetic Algorithm", *J. Systems Architecture*, vol. 47, pp. 549-554, 2001.
29. G. Attiya, Y. Hamam, "Task Allocation for Maximizing Reliability of Distributed Systems: A Simulated Annealing Approach", *J. Parallel and Distributed Computing*, vol. 66, pp. 1259-1266, 2006.
30. Y. Hamam, K. Hindi, "Assignment of Program Tasks to Processors: A Simulated Annealing Approach", *European J. Operational Research*, vol. 122, pp. 509-513, 2000.

Keywords

- **IEEE Keywords**
  - Reliability theory,
  - Distributed computing,
  - Distributed control,
  - Telecommunication network reliability,
  - Stochastic systems,
  - System performance,
  - Uncertainty,

- Stochastic processes,
- Communication networks,
- Difference equations

- **INSPEC: Controlled Indexing**

- differential equations,
- distributed processing,
- Monte Carlo methods,
- probability,
- resource allocation,
- stochastic processes,
- topology

- **INSPEC: Non-Controlled Indexing**

- service reliability,
- distributed computing system,
- random node failures,
- server nodes,
- nonzero probability,
- probabilistic framework,
- stochastic topological changes,
- heterogeneous nodes,
- stochastic service,
- communication network,
- distributed load-balancing,
- stochastic regeneration,
- difference-differential equation,
- Monte Carlo simulation

- **Author Keywords**
  - Renewal theory,
  - queuing theory,
  - reliability,
  - distributed computing,
  - load balancing.