Decentralized Load Balancing for Improving Reliability in Heterogeneous Distributed Systems

Jorge E. Pezoa*, Sagar Dhakal† and Majeed M. Hayat*‡ * Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87131, USA; E-Mail: {hayat,jpezoa}@ece.unm.edu. † Naval Research Laboratory (NRL), Washington D.C., USA; E-Mail: sagar.dhakal@gmail.com. ‡ M. M. Hayat is also with Center for High Technology Materials, University of New Mexico, Albuquerque, NM, USA.

***Abstract*—A probabilistic analytical framework for decentralized load balancing (LB) strategies for heterogeneous distributed-computing systems (DCSs) is presented with the overall goal of maximizing the service reliability in the presence of random failures. The service reliability of a DCS is defined as the probability of successfully serving a specified workload before all the computing nodes fail permanently. In the framework considered the service and failure times of nodes are random, the communication times in the network are both tangible and stochastic, and LB is performed synchronously by all the nodes during the runtime of each submitted workload. By taking a novel regenerative stochastic-analysis approach, the service reliability of a two-node DCS is characterized analytically. This formulation, in turn, is used to form and solve an optimization problem, yielding LB policies with maximal reliability. A scalable extension of the two-node formulation to an arbitrary size system is also presented. The validity of the proposed theory is studied using both Monte-Carlo simulations and real experiments on a small-scale testbed.**

***Keywords*-renewal theory, queuing theory, reliability, distributed computing, load balancing**

I. Introduction

A distributed-computing system (DCS) is an aggregate of heterogeneous and geographically dispersed computing elements (CEs) that cooperatively execute applications in a parallel fashion. In order to process a workload, a DCS first has to divide such workload into independent, indivisible units called tasks. Next, these tasks are allocated onto the computing nodes for their parallel processing. As workload and runtime conditions evolve, some CEs may find themselves overburdened with tasks awaiting to be served, while other CEs may be lightly loaded or even idle. This undesired situation of load imbalance can be solved by reallocating tasks among the nodes. Such task reallocation action is referred to in the literature as load balancing (LB). It is well-known that LB has to be intelligently executed, otherwise the utilization of the computing resources can be severely affected [1], [2].

In DCSs where no central coordinator is available, nodes have to exchange information on the number of tasks queued at each node. Such information is used by each node to estimate whether its load is balanced or not with respect to the total load in the system. In addition, failure-notice (FN) messages have to be exchanged among the nodes in order to detect which nodes have failed. However, unavoidable speed and bandwidth limitations in practical communication networks introduce uncertainty in the detection process. Therefore, nodes must take LB decisions with incomplete and/or dated knowledge on the state of the DCS.

It is known that LB can be effectively used to mitigate the effect of node failures on the execution of workloads. The goal is to devise an LB policy that maximizes the reliability of serving a workload before all the CEs fail, while the response time of such workload is simultaneously reduced [1]–[3]. To date, existing analytical solutions to this problem have been based upon multi-objective optimization approaches, for which several service reliability models have been proposed. These approaches have considered frameworks where: communication delays are either deterministic or random [1], [4], [5], task and/or hardware redundancy is used to compensate for communication costs [2], *a priori* information on the network configuration is exploited to achieve computationally fast solutions by means of simulated annealing or heuristic optimization [2], [6], [7].

In this paper we have considered the problem of devising LB strategies for maximizing the service reliability of a DCS. In our study we have considered heterogeneous DCSs, where server nodes provide random task service-times and can fail permanently at any random time. Moreover, it is assumed that the communication network imposes stochastic non-negligible transfer times, which depend on the amount of data to be transferred. A novel theory based on stochastic regeneration is developed to analytically model the service reliability of a two-node DCS by means of a set of recurrence equations. Next, we use the recurrence equations to systematically devise decentralized dynamic load balancing (DLB) policies for maximizing the service reliability. Using training data, we have estimated the parameters of a small-scale distributed computing (DC) testbed and our regenerative model was employed to predict its service reliability. This prediction is compared with both Monte-Carlo (MC) simulations and experiments conducted on the testbed. This study is motivated by our need to assess reliability in scenarios such as DC in wireless sensor networks and network resilience in harsh environments, like those resulting from stress inflicted by

Dr. Sagar Dhakal is an ASEE postdoctoral fellow in acoustic and signal processing branch at NRL. This work was conducted before Dr. Dhakal joined NRL and does not represent any view from NRL.

weapons of mass destruction.

This paper is organized as follows. In Section II we build the regeneration-based stochastic model for the service reliability in DCSs. In Section III we apply the theory to devise LB strategies that maximize the service reliability of DCSs. Their performance is tested both theoretically and experimentally and our conclusions are given in Section IV.

II. Theory

A. Problem statement

The problem addressed in this paper is concerned with maximizing the service reliability of a workload, which is defined as the probability of successfully serving a workload in a DCS whose server nodes can fail permanently with non-zero probability. The DCS is composed of n nodes and each node can communicate to any other node by exchanging messages. The workload to be executed is composed of M indivisible tasks. Let $m_j \geq 0$ be an integer representing the number of tasks queued at the jth node at time $t=0$, with $\sum_{j=1}^{n} m_j = M$. The service time of each task as well as the failure times of all functional nodes are assumed to be random. In addition, it is assumed that all the nodes are functioning at $t=0$ and that they have an estimate of the amount of tasks queued at the other nodes in the system. In order to maximize the service reliability of the workload, a decentralized, synchronous DLB action is performed at a prescribed 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 functional node. Naturally, these task exchanges over the network take random transfer times. For analytical tractability, the following assumptions are imposed on the random times that characterize the DCS.

Assumption A1: The following random variables are exponentially distributed: (i) W_{ki}: the service time of the ith task at the kth node (with rate λ_{d_k}); (ii) Y_k: the failure time of the kth node (with rate λ_{f_k}); (iii) X_{jk}: the transfer time of a FN packet sent from the jth to the kth node (with rate λ_{jk}); and (iv) Z_{ik}: the transfer time of the ith group of tasks sent to the kth node (with rate $\tilde{\lambda}_{i,k}$). In addition, we have assumed that the mean transfer time of the ith group of tasks being transferred from the jth to the kth node follows the first-order approximation: $\tilde{\lambda}_{i,k}^{-1} = a_{jk} L_{jk}^{i} + b_{jk}$, 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_{jk}^{i} is the number of tasks in the ith group.

Assumption A2: The random variables listed in Assumption A1 are mutually independent.

Assumption A3: To tolerate failures of individual nodes, each server in the DCS is equipped with a backup system that contains replicas of the tasks queued at the server. The backup systems are simple reliever equipments that do not process any task. When a node becomes faulty, the backup system broadcasts a FN packet to the DCS, and next, reallocates replicas of all the unserved tasks among the functional nodes. Also, the backup system handles the reception of tasks in transit to a faulty node by retransmitting such tasks back to the functional nodes.

Assumption A4: We assume that there is no arrival of *external tasks* to the DCS for $t>0$.

B. Decentralized dynamic load balancing policy

We have defined a decentralized DLB policy that performs the followings steps, at each node, at the LB instant:

1) Imbalance detection: The jth node computes its excess load, denoted as $L_j^{ex}(t_b)$, using the formula

$$L_j^{ex}(t_b) = Q_j(t_b) - \frac{\lambda_{f_j}^{-1}}{\sum_{l=1}^{n} \lambda_{f_l}^{-1}} \sum_{l=1}^{n} \hat{Q}_{l,j}(t_b), \qquad (1)$$

where $Q_j(t_b)$ is the number of tasks queued at the jth node at time t_b, $\hat{Q}_{l,j}(t_b)$ is the number of tasks queued at the lth node as perceived by the jth node at $t=t_b$. If the excess load is positive, then the node considers itself as imbalanced. Note that last term in (1) is the fraction of the total load that the jth node should process if the relative reliability of the nodes is used as the allocation criterion.

2) Selection of recipient nodes: If the jth node is imbalanced, we define the collection $\mathcal{U}_j$ of candidate receiver nodes as all those nodes that, at the balancing instant, are perceived by the imbalanced node as underloaded with respect to their own perceived averages of the total workload; namely, $\mathcal{U}_j \triangleq \{k : L_{k,j}^{ex}(t_b) < 0\}$, where $L_{k,j}^{ex}(t_b)$ is the excess load at the node k as perceived by the node j and is defined as $L_{k,j}^{ex}(t_b) = \hat{Q}_{k,j}(t_b) - \lambda_{f_j}^{-1} (\sum_{l=1}^{n} \lambda_{f_l}^{-1})^{-1} \sum_{l=1}^{n} \hat{Q}_{l,j}(t_b)$.

3) Load partitioning: The jth imbalanced node partitions its excess load among all those nodes that are estimated to be below the average load of the DCS and the partition p_{jk} is defined as $p_{jk} \triangleq L_{k,j}^{ex}(t_b) / (\sum_{l \in \mathcal{U}_j} L_{l,j}^{ex}(t_b))$ whenever $k \in \mathcal{U}_j$ and 0 otherwise. In the presence of stochastic transfer times, partitions have to be adjusted in order to account for the uncertainty introduced by the random communication times. Here, we consider that load to be transferred from the jth to the kth node is adjusted according to what is called the *load-balancing gain* [8], [9], which is denoted as K_{jk}, yielding $L_{jk}(t_b) = \lfloor K_{jk} p_{jk} L_j^{ex}(t_b) \rfloor$, with $\lfloor x \rfloor$ the greatest integer less than or equal to x. Note that the quantity $p_{jk} L_j^{ex}$ is fixed and defines the maximum number of tasks to be reallocated from node j to k; it is assumed here that the LB gains are rational numbers in the interval [0,1]. Note also that LB gains are parameters that have to be optimized in order to maximize the service reliability.

C. Service reliability model for a two-node DCS

1) State-space representation: At any time, the DCS can be described by the following quantities: (i) the number of tasks queued at each node; (ii) the functional or dysfunctional state of each node in the system; and (iii) the number of tasks in transit over the communication network. Based upon these quantities, we have constructed a discrete-state continuous-time model for the service reliability.

For a two-node DCS and for each instant of time, we assign an ordered pair, denoted as $\mathbf{M}$, for describing the number of tasks queued at each node. The first component of the vector

contains the number of tasks queued at the first node while the second corresponds to the number of tasks queued at the second node at time t. Similarly, we define for each instant of time the *system-function state* as the binary vector $\mathbf{F}$. At time t, if the jth component of $\mathbf{F}$ is "1" (correspondingly, "0") this indicates that the jth node is functioning (correspondingly, faulty).

Due to the reallocation of tasks among the nodes, some tasks are queued in the communication network during their transfer time. Let $g_k \in \{0,1,2,3\}, k=1,2$, represent the number of different groups of tasks that are simultaneously in transit to the kth node at time t. We assign the vector $\mathbf{c}_k$, of size g_k+1, such that first component of $\mathbf{c}_k$ is always set to g_k, while the remaining components indicate the number of tasks per group that are in transit to the kth node. We now define the *network state* as the concatenated vector $\mathbf{C} \triangleq (\mathbf{c}_1, \mathbf{c}_2)$. For example, $\mathbf{C} = ([2\ 5\ 3],[1\ 4])$ at time t corresponds to the network state for which two different groups of tasks (five tasks in the first group and three tasks in the second group) are being transferred to the first node ($\mathbf{c}_1 = [2\ 5\ 3]$), while one group of four tasks is being transferred to the second node ($\mathbf{c}_2 = [1\ 4]$).

According to the decentralized DLB policy, at the balancing instant the jth node must transfer $L_{jk} = \lfloor K_{jk} p_{jk} L_j^{ex} \rfloor$ tasks to the kth functional node, while $r_j = m_j - L_{jk}$ tasks remain queued at the jth node. Let $T_{\mathbf{M}}^{\mathbf{F}}(\mathbf{K};\mathbf{C})$ be the time to serve all the tasks in the DCS when the initial system state at $t=t_b$ is specified by $\mathbf{M} = (r_1, r_2)$, $\mathbf{F} = (1,1)$ and $\mathbf{C} = ([1\, L_{21}],[1\, L_{12}])$, and $\mathbf{K} = [K_{12}\ K_{21}]$. Note that the time to serve depends upon the LB policy. Our objective is to calculate the *service reliability* of the DCS, i.e. the probability of serving an entire workload before all the CEs fail. We denote the service reliability as $R_{\mathbf{M}}^{\mathbf{F}}(\mathbf{K};\mathbf{C}) \triangleq \mathsf{P}\{T_{\mathbf{M}}^{\mathbf{F}}(\mathbf{K};\mathbf{C}) < \infty\}$.

2) Regeneration time and regenerative characterization of the service reliability: We will undertake an approach based on the concept of stochastic regeneration to obtain recurrence equations that characterize the service reliability. Our key idea is to introduce a special random variable, called the *regeneration time*, τ, which is defined as the minimum of the following four 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 FN at any node, or the time to the *first* arrival of a group of tasks at any node. The key property of this regeneration time is that upon the occurrence of the regeneration event $\{\tau = s\}$, a *fresh* copy of the original stochastic process [from which $T_{\mathbf{M}}^{\mathbf{F}}(\mathbf{K};\mathbf{C})$ is defined] will emerge at time s with a *new* initial system configuration that transpires from the regeneration event.

More precisely, the occurrence of the regeneration event $\{\tau = s\}$ gives birth to a new DCS at $t=s$ whose random times satisfy Assumptions A1 and A2 while having its own initial system configuration. The new initial configuration can be one of the following: (i) a new initial task distribution if the regeneration event is a service to a task; (ii) new $\mathbf{F}$ and $\mathbf{C}$ states if the regeneration event is a node failure; (iii) a new $\mathbf{F}$ state if the regeneration event is the arrival of a FN packet; or (iv) a new task distribution and a new $\mathbf{C}$ state if the regeneration event is the arrival of a group of tasks.

Our main result, Theorem 1, characterizes the reliability in the form of a set of coupled difference equations.

Theorem 1: For r_1, r_2, L_{jk}^i non-negative integers, the service reliability, $R_{\mathbf{M}}^{\mathbf{F}}(\mathbf{K};\mathbf{C})$, satisfies the recursions:

$$R_{(r_1,r_2)}^{(1,1)}(\mathbf{K};\mathbf{C}_{g_1,g_2}) = \lambda_A^{-1}\Big(\lambda_{d_1} R_{(r_1-1,r_2)}^{(1,1)}(\mathbf{K};\mathbf{C}_{g_1,g_2}) + \lambda_{d_2} R_{(r_1,r_2-1)}^{(1,1)}(\mathbf{K};\mathbf{C}_{g_1,g_2}) + \lambda_{f_1} R_{(0,r_2)}^{(0,1)}(\mathbf{K};\mathbf{C}_{g_1,g_2+1}) + \lambda_{f_2} R_{(r_1,0)}^{(1,0)}(\mathbf{K};\mathbf{C}_{g_1+1,g_2}) + \sum_{i=1}^{g_1} \tilde{\lambda}_{i,1} R_{(r_1+L_{21}^i,r_2)}^{(1,1)}(\mathbf{K};\mathbf{C}_{g_1-i,g_2}) + \sum_{i=1}^{g_2} \tilde{\lambda}_{i,2} R_{(r_1,r_2+L_{12}^i)}^{(1,1)}(\mathbf{K};\mathbf{C}_{g_1,g_2-i})\Big), \quad (2)$$

$$R_{(r_1,0)}^{(1,0)}(\mathbf{K};\mathbf{C}_{g_1,g_2}) = \lambda_B^{-1}\Big(\lambda_{d_1} R_{(r_1-1,0)}^{(1,0)}(\mathbf{K};\mathbf{C}_{g_1,g_2}) + \sum_{i=1}^{g_1} \tilde{\lambda}_{i,1} R_{(r_1+L_{21}^i,0)}^{(1,0)}(\mathbf{K};\mathbf{C}_{g_1-i,g_2}) + \sum_{i=1}^{g_2} \tilde{\lambda}_{i,2} R_{(r_1,0)}^{(1,0)}(\mathbf{K};\mathbf{C}_{g_1+i,g_2-i})\Big), \quad (3)$$

$$R_{(0,r_2)}^{(0,1)}(\mathbf{K};\mathbf{C}_{g_1,g_2}) = \lambda_C^{-1}\Big(\lambda_{d_2} R_{(0,r_2-1)}^{(0,1)}(\mathbf{K};\mathbf{C}_{g_1,g_2}) + \sum_{i=1}^{g_1} \tilde{\lambda}_{i,1} R_{(0,r_2)}^{(0,1)}(\mathbf{K};\mathbf{C}_{g_1-i,g_2+i}) + \sum_{i=1}^{g_2} \tilde{\lambda}_{i,2} R_{(0,r_2+L_{12}^i)}^{(0,1)}(\mathbf{K};\mathbf{C}_{g_1,g_2-i})\Big), \quad (4)$$

$$R_{(r_1,0)}^{(1,0)}(\mathbf{K};\mathbf{C}_{0,0}) = (\lambda_{d_1}(\lambda_{d_1}+\lambda_{f_1})^{-1})^{r_1}, \quad (5)$$

$$R_{(0,r_2)}^{(0,1)}(\mathbf{K};\mathbf{C}_{0,0}) = (\lambda_{d_2}(\lambda_{d_2}+\lambda_{f_2})^{-1})^{r_2}, \quad (6)$$

where $\tilde{\lambda}_{i,k}^{-1} = a_{jk} L_{jk}^i + b_{jk}$, $\lambda_A = \sum_{k=1}^{2} \lambda_k^{'} + \sum_{i=1}^{g_1} \tilde{\lambda}_{i,1} + \sum_{i=1}^{g_2} \tilde{\lambda}_{i,2}$, $\lambda_B = \lambda_1^{'} + \lambda_{21} + \sum_{i=1}^{g_1} \tilde{\lambda}_{i,1} + \sum_{i=1}^{g_2} \tilde{\lambda}_{i,2}$, $\lambda_C = \lambda_2^{'} + \lambda_{12} + \sum_{i=1}^{g_1} \tilde{\lambda}_{i,1} + \sum_{i=1}^{g_2} \tilde{\lambda}_{i,2}$, $\lambda_k^{'} = \lambda_{d_k} + \lambda_{f_k}$, $\mathbf{C}_{g_1,g_2} = ([g_1\, L_{21}^1 \ldots L_{21}^{g_1}],[g_2\ L_{12}^1 L_{12}^{g_2}])$, $\mathbf{C}_{g_1+1,g_2} = ([g_1\ +\ 1\ L_{21}^1 \ldots\ L_{21}^{g_1}\ r_2],[g_2\, L_{12}^1 \ldots L_{12}^{g_2}])$, $\mathbf{C}_{g_1,g_2+1} = ([g_1\ L_{21}^1 \ldots L_{21}^{g_1}],[g_2+\ \ 1\, L_{12}^1 \ldots L_{12}^{g_2}\, r_1])$, $C_{g_1-i,g_2} = ([g_1\ -\ \ 1\ L_{21}^1 \ldots L_{21}^{i-1}\ L_{21}^{i+1} \ldots L_{21}^{g_1}],[g_2\ L_{12}^1 \ldots L_{12}^{g_2}])$, $C_{g_1,g_2-i} = ([g_1\ L_{21}^1 \ldots L_{21}^{g_1}],[g_2 - 1 L_{12}^1 \ldots L_{12}^{i-1}\ L_{12}^{i+1} \ldots L_{12}^{g_2}])$, $\mathbf{C}_{g_1+i,g_2-i} = ([g_1\ +\ 1\ L_{21}^1 \ldots L_{21}^{g_1}\ L_{12}^i],[g_2\ -\ 1\ L_{12}^1 \ldots L_{12}^{i-1}\ L_{12}^{i+1} \ldots L_{12}^{g_2}])$, and $\mathbf{C}_{g_1-i,g_2+i} = ([g_1 - 1\ L_{21}^1 \ldots L_{21}^{i-1} L_{21}^{i+1} \ldots L_{21}^{g_1}],[g_2 + 1\ L_{12}^1 \ldots L_{12}^{g_2}\ L_{21}^i])$. Also, we trivially have that the service reliability is equal to one, for any state $\mathbf{F}$, if there are no tasks to be served in the DCS. Also, if at least one task is queued in the DCS and both nodes have failed, the reliability is equal to zero.

Proof: First, note that from Assumptions A1 and A2 it is straightforward to show that τ follows an exponential distribution with parameter λ_A. Next, the key idea is to condition the probability of serving the workload on τ, and then average the conditional probability with respect to the distribution of τ. So, the probability of serving the workload for the initial state $\mathbf{M} = (r_1\, r_2)$, $\mathbf{F} = (1,1)$ and $\mathbf{C} = ([1\, L_{21}]\,[1\, L_{12}])$ can be written as

$$R_{(r_1,r_2)}^{(1,1)}(\mathbf{K};\mathbf{C}) = \int_0^{\infty} \mathsf{P}\{T_{(r_1,r_2)}^{(1,1)}(\mathbf{K};\mathbf{C}) < \infty | \tau = s\}\, f_\tau(s)\, ds.$$

Moreover, we can further condition the first integrand at the right side of the latter equation on all the possible, disjoint

regeneration events occurring at the time $\tau = s$ as

$$\begin{aligned}
&\mathsf{P}\{T^{(1,1)}_{(r_1,r_2)}(\mathbf{K};\mathbf{C})<\infty|\tau=s\}\\
&=\textstyle\sum_{k=1}^{2}\mathsf{P}\{T^{(1,1)}_{(r_1,r_2)}(\mathbf{K};\mathbf{C})<\infty|\tau=s,\tau=W_{k1}\}\\
&\times\mathsf{P}\{\tau=W_{k1}|\tau=s\}+\textstyle\sum_{k=1}^{2}\mathsf{P}\{\tau=Z_{1k}|\tau=s\}\\
&\times\mathsf{P}\{T^{(1,1)}_{(r_1,r_2)}(\mathbf{K};\mathbf{C})<\infty|\tau=s,\tau=Z_{1k}\}\\
&+\textstyle\sum_{j=1}^{2}\mathsf{P}\{T^{(1,1)}_{(r_1,r_2)}(\mathbf{K};\mathbf{C})<\infty|\tau=s,\tau=Y_{j}\}\\
&\times\mathsf{P}\{\tau=Y_j|\tau=s\}. \qquad (7)
\end{aligned}$$

Now, consider for instance the regeneration event $\{\tau=s,\tau=W_{11}\}$. This is precisely the service of the first task at the first node before any other activity takes place in the DCS. Thus, upon the occurrence of $\{\tau=s,\tau=W_{11}\}$, the state vectors remain the same, i.e., $\mathbf{F}=(1,1)$ and $\mathbf{C}=([1\,L_{21}]\,[1\,L_{12}])$, while r_1-1 tasks are now queued at the node 1 and r_2 are queued at node 2. Therefore, by construction $\mathsf{P}\{T^{(1,1)}_{(r_1,r_2)}(\mathbf{K};\mathbf{C}) < \infty|\tau = s,\tau = W_{11}\} = \mathsf{P}\{\tau+T'^{(1,1)}_{(r_1-1,r_2)}(\mathbf{K};\mathbf{C}) < \infty|\tau = s,\tau = W_{11}\}$, where $T'^{(1,1)}_{(r_1-1,r_2)}(\mathbf{K};\mathbf{C})$ is the random service time taken by the new DCS that has emerged at time τ and is described by the same state vectors $\mathbf{F}$ and $\mathbf{C}$, while r_1-1 and r_2 tasks are queued at the first and second node, respectively. It can be shown that conditional on the regeneration event $\{\tau = s,\tau = W_{11}\}$ and due to the memoryless property of the exponential distribution, the random times emerging at the regeneration time satisfy assumptions A1 and A2. Namely, the service of a task at any node, the failure time at any node, and the transfer times of either FN packets or groups of tasks, satisfy these assumptions. Therefore, we can shift the time origin from $t = 0$ to $t = s$ without changing the statistics of the random times. This leads us to the following result: $\mathsf{P}\{\tau + T'^{(1,1)}_{(r_1-1,r_2)}(\mathbf{K};\mathbf{C}) < \infty|\tau = s,\tau = W_{11}\} = \mathsf{P}\{s + T^{(1,1)}_{(r_1-1,r_2)}(\mathbf{K};\mathbf{C}) < \infty\} = \mathsf{P}\{T^{(1,1)}_{(r_1-1,r_2)}(\mathbf{K};\mathbf{C}) < \infty\}$. Similar results can be obtained for the other regeneration events by considering the effects that the regeneration event has on the system configuration. For example, $\mathsf{P}\{T^{(1,1)}_{(r_1,r_2)}(\mathbf{K};\mathbf{C}) < \infty|\tau = s,\tau = Z_{11}\} = \mathsf{P}\{T^{(1,1)}_{(r_1+L_{21},r_2)}\big(\mathbf{K};([0],[1\,L_{12}])\big) < \infty\}$ and $\mathsf{P}\{T^{(1,1)}_{(r_1,r_2)}(\mathbf{K};\mathbf{C}) < \infty|\tau = s,\tau = Y_1\} = \mathsf{P}\{T^{(0,1)}_{(0,r_2)}\big(\mathbf{K};([1\,L_{21}],[2\,L_{12}\,r_1])\big)<\infty\}$. We refer the reader to [10] for a proof of these two results.

In addition, it is not hard to show, see [10], that for any s: (i) $\mathsf{P}\{\tau = W_{k1}|\tau = s\} = \lambda_{d_k}\lambda_A^{-1}$; (ii) $\mathsf{P}\{\tau = X_{jk}|\tau = s\} = \lambda_{jk}\lambda_A^{-1}$; and (iii) $\mathsf{P}\{\tau = Y_k|\tau = s\} = \lambda_{f_k}\lambda_A^{-1}$. Thus, (7) becomes

$$\begin{aligned}
&R^{(1,1)}_{(r_1,r_2)}(\mathbf{K};\mathbf{C}) = \lambda_A^{-1}\int_0^{\infty}\Big(\lambda_{d_1}R^{(1,1)}_{(r_1-1,r_2)}(\mathbf{K};\mathbf{C})\\
&+\lambda_{d_2}R^{(1,1)}_{(r_1,r_2-1)}(\mathbf{K};\mathbf{C})+\tilde{\lambda}_{1,1}R^{(1,1)}_{(r_1+L21,r_2)}\big(\mathbf{K};([0],[1\,L_{12}])\big)\\
&+\tilde{\lambda}_{1,2}R^{(1,1)}_{(r_1,r_2+L21)}\big(\mathbf{K};([1\,L_{21}],[0])\big)\\
&+\lambda_{f_1}R^{(0,1)}_{(0,r_2)}\big(\mathbf{K};([1\,L_{21}],[2\,L_{12}\,r_1])\big)\\
&+\lambda_{f_2}R^{(1,0)}_{(r_1,0)}\big(\mathbf{K};([2\,L_{21}\,r_2],[1\,L_{12}])\big)\Big)\;f_\tau(s)\;ds. \qquad (8)
\end{aligned}$$

By noting both that terms in parenthesis do not depend on s, and that the resulting integral is one, we obtain (2) for the initial state $\mathbf{M}=(r_1\,r_2)$, $\mathbf{F}=(1,1)$ and $\mathbf{C}=([1\,L_{21}]\,[1\,L_{12}])$. The remaining equations are obtained applying the same principles for other initial states. □

Note that equation (2) models the service reliability of a DCS in scenarios where both nodes are functioning and tasks are being transferred in the communication network. Equations (3) and (4) characterize the reliability for cases when only one node is functioning and tasks are queued in the network. Finally, (5) and (6) provide expressions for the service reliability in cases where only one node is working and all the tasks are queued at the working node.

3) Optimization problem: Recurrence equations in Theorem 1 completely characterize the reliability of a two-node DCS. Using this model, we can formulate an optimization problem whose solution specifies the optimal number of tasks to reallocate among the nodes and yields the maximal service reliability of a two-node DCS. Mathematically,

$$(K_{12}^*,K_{21}^*) = \underset{(K_{12},K_{21})}{\operatorname{argmax}}\; R^{(1,1)}_{(r_1,r_2)}\big(\mathbf{K};([1\,L_{21}],[1\,L_{12}])\big), \qquad (9)$$

subject to: K_{12}, K_{21} are numbers in [0,1].

D. Algorithmic extension for multi-node DCSs

In order to provide a scalable and efficient solution for the case of a multi-node DCS, we follow [10] and extend our solution by means of an algorithm that decomposes an n-node system in several two-node DCSs, and invokes Theorem 1 every time load has to be exchanged between pairs of nodes. Our algorithm to devise DLB policies for multi-node DCSs is described by the following steps.

Initialization: Let the jth node be the sender node and let U_j be its the collection of candidate receiver nodes. Let $K_{jk}=1$ denote the initial value for the LB gain associated with the load transfer from the jth to the kth node. (Namely, we have assumed that the jth node can send full load partitions to the recipient nodes.) Let $\mathcal{U}'_j$ be the set denoting those recipient nodes k, for which the pairwise optimal LB gain, K^*_{jk}, has been already calculated.

Step 1: Select the first recipient node, say the lth node, and we consider a two-node system comprising nodes l and j. As a result, upon the execution of LB at t_b, the lth and the jth nodes will have loads $\hat{Q}_{l,j}(t_b)$ and $Q_j(t_b) - (\sum_{i\in(\mathcal{U}\setminus\{l\})}\lfloor K_{ji}p_{ji}L^{ex}_j(t_b)\rfloor + \lfloor K_{jl}p_{jk}L^{ex}_j(t_b)\rfloor)$, respectively, while $\lfloor K_{jl}p_{jk}L^{ex}_j(t_b)\rfloor$ tasks are assumed to be in transit from the jth node to the lth node. Theorem 1 can now be invoked to compute K^*_{jl}. Next, we update the set $\mathcal{U}'_j$ by including the lth node in it: $\mathcal{U}'_j = \{l\}$.

Step k: Select another recipient node, say the kth node, from the collection $\mathcal{U}_j \setminus \mathcal{U}'_j$. Then, we employ $K_{ji} = 1$ for all $i \in \mathcal{U}_j \setminus (\{k\} \cup \mathcal{U}'_j) = \mathcal{V}$ as well as K^*_{js} for all $s \in \mathcal{U}'_j$, while K^*_{jk} is obtained by considering a two-node system with: $\hat{Q}_{k,j}(t_b)$ and $Q_j(t_b) - (\sum_{i\in\mathcal{V}}\lfloor p_{ji}L^{ex}_j(t_b)\rfloor + \sum_{i\in\mathcal{U}'_j}\lfloor K^*_{ji}p_{ji}L^{ex}_j(t_b)\rfloor + \lfloor K_{jk}p_{jk}L^{ex}_j(t_b)\rfloor)$ tasks at the nodes k and j, respectively, while $\lfloor K_{jk}p_{jk}L^{ex}_j(t_b)\rfloor$ tasks are in transit from the jth to the kth node. We next augment the set $\mathcal{U}'_j$ by including the kth node in it.

Repeat: The kth step is repeated until the gains of all the nodes in $\mathcal{U}_j$ are calculated, that is until $\mathcal{U}'_j = \mathcal{U}_j$.

Termination condition: We iterate over the previous steps to compute all the LB gains until either all of them converge to a certain value or a predefined maximum number of iterations, κ_{max}, is executed. The announced suboptimal LB gains are those obtained after either one of these two termination conditions are met.

III. Maximizing service reliability

A. 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 and the communication network. As the occurrence of a failure at any node is simulated by software, the backup systems are processes that handle the tasks upon failure. The communication network employed in our architecture is the Internet, where the terminal communication links are either wired or wireless. For some links we have introduced artificial latency using traffic shaper applications that reduce network interfaces' speed to slow rates such as 512 Kbps.

The software architecture of the DCS is divided in three layers: application, task reallocation and communication. The application layer executes matrix multiplications, which is the application selected to illustrate a computing intensive distributed application. We have defined a task as the multiplication of one row by a static matrix, which is replicated in all the nodes. To achieve variability in the processing speed of the nodes, randomness is introduced in the size of each row by independently choosing its arithmetic precision with an exponential distribution. In addition, the application layer performs two extra tasks: estimates the queue length of the other nodes and simulates failures at the nodes. The task reallocation layer executes the LB policy and computes the optimal LB gains using (2)–(6). In addition, upon the failure of a node, this layer executes the reallocation of tasks as is described in Assumption A3. Finally, the communication layer handles the transfer of tasks as well as the periodic exchange of queue-length information (QI) packets among the nodes. This layer also creates and transfers FN packets among the nodes. Each node uses the UDP protocol to send QI and FN packets to the other nodes, while the TCP protocol is used to transfer tasks between the nodes.

B. Maximal service reliability of a two-node DCS

Example 1: We have assessed the service reliability of a DCS composed of two nodes that are separated by a large geographic distance and communicate through the Internet. The initial workload and the average failure times, were set to be: $m_1 = 50$ tasks and $m_2 = 25$ tasks, $\lambda_{f_1}^{-1} = 300$ s and $\lambda_{f_2}^{-1} = 100$ s. The remaining system parameters were estimated from data obtained after conducting training experiments on the two-node DCS. The estimated service rates of each node are $\lambda_{d_1} = 0.1682$ tasks per second (tps) and $\lambda_{d_2} = 0.4978$ tps. The estimated mean arrival times of each FN packet are $\lambda_{12}^{-1} = 4.6402$ s and $\lambda_{21}^{-1} = 1.6659$ s. Finally, the estimated parameters 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). Fig. 1a shows the empirical probability distribution function (pdf) of the service time of node 2 as well as its exponentially approximated pdf. Fig. 1b shows that the average transfer times grows linearly with the number of tasks.

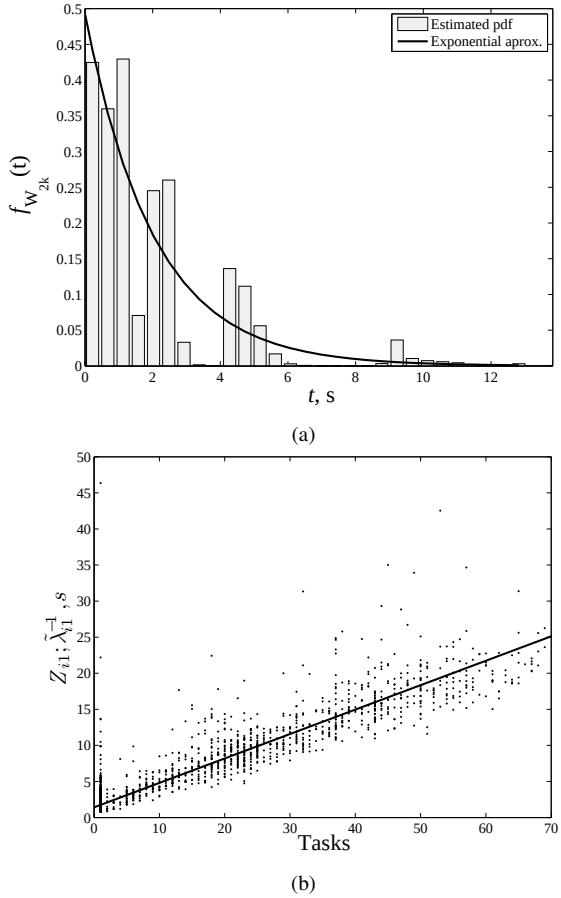

Fig. 1. (a) Empirical pdf of service time at node 2 and its exponential approximation. (b) Realizations of task transfer-time and straight-line approximation for the mean task transfer-time.

Let us first look at the solution to the recurrence equations for the initial state $\mathbf{F} = (1, 1)$ and $\mathbf{C} = ([1\, L_{21}], [\, L_{21}])$. In order to explore all the possible amounts of tasks to exchange among the nodes we have defined the number of tasks to transfer from the jth to the kth nodes to be: $L_{jk} = \lfloor K_{jk} m_j \rfloor$. In Fig. 2a the service reliability under two choices of K_{21} is plotted as a function of K_{12}. In addition to the theoretical predictions, Fig. 2a shows MC simulations as well as experimental results obtained for the decentralized DLB policies. In our simulations, each success probability is calculated by averaging outcomes (failures or successes) from 4000 independent realizations of each policy, while for the experiments the success probability is calculated by averaging the results of 500 independent trials for each policy. The upper plot in Fig. 2a corresponds to the case when $K_{21} = 0.05$ while the lower plot is for $K_{21} = 0.50$. Note that for small values of K_{12}, node 1 (slower) keeps most of its initial load. Consequently, the load distribution remains unfair even after LB is performed. Therefore, the time required to serve the workload

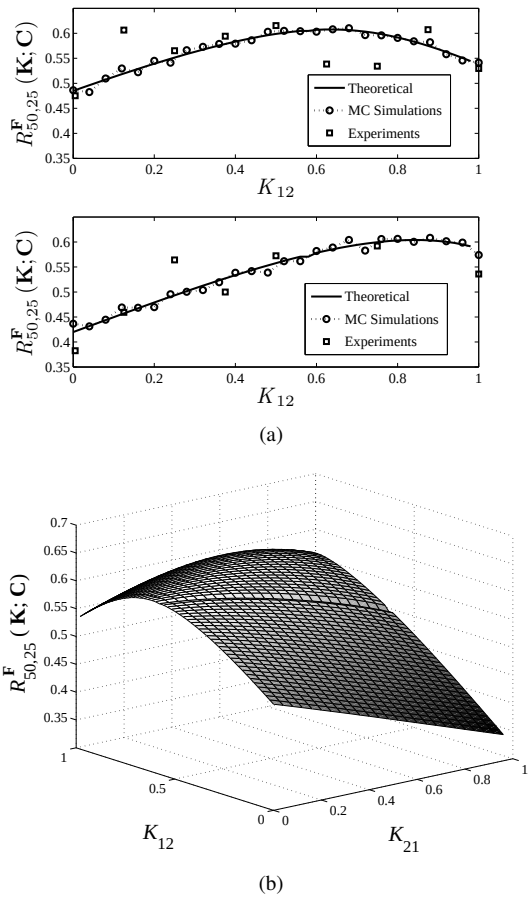

Fig. 2. Service reliability as a function of: (a) LB gain of the slowest node, and (b) both LB gains.

increases and, as a consequence, the service reliability is reduced. When K_{12} approaches to 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; thereby, the service reliability is reduced by an excessive queuing of tasks in the communication network. Note that simulation results are in good agreement to our theoretical predictions, and remarkably, experimental results show a fairly good agreement with the theoretical curves as well. Next, we look at the solution of the equations in Theorem 1. Figure 2b shows theoretical predictions for the service reliability as a function of both LB gains. Note that an improper selection of the LB gains can produce a significant reduction on the service reliability, as is depicted for the case of choosing $K_{12} = 0.1$ and $K_{21} = 0.6$. Finally, by solving the optimization problem defined in Section II-C3 we obtain the optimal DLB policy: (K_{12}^*, K_{21}^*)=(0.6,0), which provides a theoretical optimal service reliability of 0.601. For such policy, the service reliability estimated using 4000 independent simulations is 0.613, while for 500 independent experiments conducted on the testbed the estimated reliability is 0.62.

C. Maximizing the service reliability of a multi-node DCS

Now, we maximize the reliability of a five-node DCS serving a workload of $M = 150$ tasks. We have assumed that the average failure times of the nodes are $\lambda_{f_1}^{-1} = 350$ s, $\lambda_{f_2}^{-1} = 10$ s, $\lambda_{f_3}^{-1} = 20$ s, $\lambda_{f_4}^{-1} = 200$ s, and $\lambda_{f_5}^{-1} = 300$ s. The service rates, estimated using training sets of tasks on our testbed, are $\lambda_{d_1} = 0.1682$ tps, $\lambda_{d_2} = 0.4978$ tps, $\lambda_{d_3} = 0.2587$ tps, $\lambda_{d_4} = 0.2536$ tps, and $\lambda_{d_5} = 0.1836$ tps. For brevity, we provide only the minimum and the maximum values of the remaining parameters. In the case of the estimated mean arrival times of FN packets $\min_{j,k} \lambda_{jk}^{-1} = 0.343$ s and $\max_{j,k} \lambda_{jk}^{-1} = 7.727$ s, while the extreme values for the first-order approximation of the average transfer time of tasks are $\min_{j,k} a_{jk} = 0.219$ and $\max_{j,k} a_{jk} = 0.898$ in seconds per task, and $\min_{j,k} b_{jk} = 1.667$ and $\max_{j,k} b_{jk} = 7.604$.

Example 2: We devise and discuss three LB policies: (i) The *Null LB policy* where all the LB gains employed by the policy are equal to zero; (ii) The *Full LB policy* where all the LB gains are equal to one; and (iii) The *Maximal-Service LB policy* where the LB gains are computed using the algorithm presented in Section II-D. Note that the Null LB policy determines the reliability provided by the fault-tolerance mechanism described in Assumption A3. Consequently, such policy establishes the minimal service reliability that can be demanded to any effective LB policy acting on the DCS. Additionally, we have conducted a MC-based exhaustive search, over the LB gains, in order to estimate the optimal service reliability for each case considered.

Theoretical predictions obtained for the LB polices under study, and for different initial task allocations, are listed in Table I. The service reliability was obtained through simulations and values listed in Table I 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 "Exp." presents results obtained after averaging 500 realization of experiments conducted on our testbed. The first five rows of Table I represent cases where the system is totally imbalanced. Rows six, seven, eight, and nine represent cases where load is initially allocated uniformly, according to nodes' reliability, according to nodes' processing speed, and arbitrarily, respectively.

It can be observed from Table I that the Maximal-Service LB policy outperforms the other two policies in all the cases considered. In addition, the Maximal-Service policy achieves a service reliability within 80% of the optimal service reliability for each case, and in fact, the optimum is achieved in some cases. We can also note that the Maximal-Service LB policy effectively increases the service reliability provided by fault-tolerance mechanism of the DCS. This increment in the reliability is attributed to two issues: (i) the policy trades off network queuing times and node idle times by computing appropriate LB gains; and (ii) the 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.

Consider the case where all the tasks are queued at the fourth node (fourth row in Table I). If no LB action is executed,

then on an average after 200 s the fourth node fails, while on an average the following events have occurred in the DCS: (i) the second and third nodes have failed; (ii) the fourth node has been informed about the failures of the second and third nodes; and (iii) 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 node. So, we clearly note that the first and the fifth node have remained idle for long periods of time, and even worst, we notice that the second and third node were never employed to serve any task. On the contrary, if the Full LB policy is used, the fourth node decides to transfer 59, 1, 3, and 50 tasks to the first, second, third, and fifth node, respectively, while 37 tasks remain queued at the fourth node. We can deduce from the previous discussion why the Full LB policy is advantageous over the Null LB policy, as evidenced by the service reliability shown in Table I.

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: $K^*_{41} = 0.661$, $K^*_{42} = K^*_{43} = 1$, and $K^*_{45} = 0.824$; this implies that the fourth node has to transfer 38, 1, 3, and 41 tasks to the first, second, third, and fifth node, respectively. Note that by sending fewer tasks to the first and fifth node, 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 an average: (i) the fourth node is able to process only 50 tasks before it fails; and (ii) by the average failure time of the fourth node, the first and fifth node are still busy serving the tasks reallocated at the balancing instant. Therefore, the upon failure task reallocation performed by the fourth node does not introduce any idle time in the receiving nodes.

TABLE I
SERVICE RELIABILITY UNDER DIFFERENT LB POLICIES.

Initial load $(m_1, \ldots, m_5)$	Service Reliability				
	Null	Full	Max-Service		Opt.
	Theo.	Theo.	Theo.	Exp.	Theo.
(150,0,0,0,0)	0.083	0.228	0.233	0.24	0.263
(0,150,0,0,0)	0.251	0.238	0.285	0.30	0.290
(0,0,150,0,0)	0.234	0.209	0.286	0.27	0.286
(0,0,0,150,0)	0.199	0.231	0.291	0.27	0.297
(0,0,0,0,150)	0.173	0.247	0.254	0.30	0.312
(30,30,30,30,30)	0.296	0.249	0.295	0.30	0.317
(59,2,4,34,51)	0.257	0.257	0.257	0.25	0.318
(18,55,29,27,21)	0.294	0.253	0.294	0.30	0.314
(40,15,40,35,20)	0.272	0.255	0.286	0.28	0.296

It must be noted that caution has to be exercised in selecting the number of tasks to reallocate at the balancing time; otherwise, one 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. As a consequence of selecting the Full LB policy, the fourth node remains idle for 55 s before it fails. On the contrary, if the Null LB policy is employed 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 $t = 200$ s, and its average idle time is reduced to 24 s. In light of this discussion, we can now comprehend the counterintuitive behavior shown in Table I. It can be noted also 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, the DCS can exploit both the computing power of the unreliable servers and the task reallocation action executed by the backup systems upon failure of those unreliable nodes.

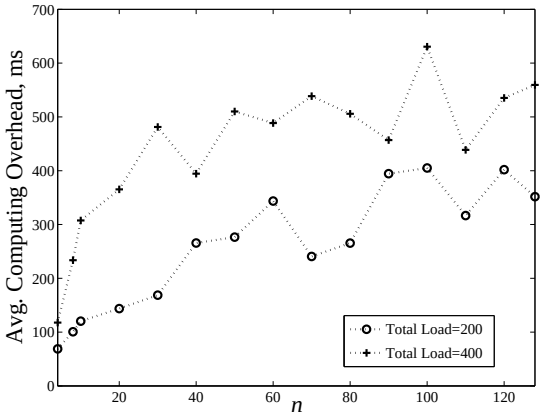

Fig. 3. Average computing time taken by the LB algorithm as a function of the number of nodes.

Finally, by means of simulations we have studied the scalability of our DLB algorithm. As our approach is decentralized, we focus only on the computations performed by a single node. We have considered the most computationally intensive case, namely, when a node has to partition its excess load among all the other nodes in the DCS. Fig. 3 shows the average computing overhead introduced by our algorithm, i.e., the execution time taken by the multi-node algorithm to compute all the LB gains, as a function of the number of nodes in the DCS, for two different amounts of tasks allocated at the sender node. The mean computing overhead was computed by averaging 100 executions of the algorithm. Roughly speaking, Fig. 3 shows that the overhead increases linearly with the number of nodes in the DCS. In fact, we can say that the overhead increases at most linearly with the number of nodes, since equations (2)–(6) have to be at most $\kappa_{max}(n-1)$ times to compute all the LB gains.

IV. CONCLUSIONS

We have presented an analytical, probabilistic framework to devise decentralized DLB polices that maximize the service reliability of heterogeneous DCSs in the presence of communication and node uncertainty. When LB is performed as dictated by our policies, 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 nodes' reliability but disregard the communication costs over the network. Moreover, our suboptimal approach to compute the LB parameters achieves a service reliability within 80% of the optimal service reliability, and in cases achieves the optimal value. Using our framework, we have predicted the service reliability of a small-scale testbed DCS and devised LB policies to enhance its reliability. Such policies

were implemented in our testbed and experimental results were compared to theoretical predictions. Our evaluations have shown not only an improvement in the service reliability, but also the remarkable accuracy of our predictions. In terms of scalability, by construction our algorithm scales linearly with the number of nodes in the DCS. Our evaluations show that the overhead in computing the LB gains for a 128 node DCS takes approximately 25% of the average service time of a task at the fastest node.

The framework for devising LB policies relies on our rigorous stochastic model for the service reliability of a DCS. This model takes into account the heterogeneity in the computing resources, the random communication and transfer delays in the network and the uncertainty associated with the number of functional nodes in the DCS. A regeneration argument has been established yielding a set of coupled recurrence equations that analytically characterize the service reliability of a two-node DCS.

We remark that one can further extend the theory presented here to optimally determine when the LB action must be performed. Such extension requires to augment the number of states of the system by including a new state vector, whose goal is to describe if nodes are informed or not about the queue-length of the other nodes in the DCS.

ACKNOWLEDGMENT

This work was supported by the Defense Threat Reduction Agency (Combating WMD Basic Research Program).

REFERENCES

[1] Y.-S. Dai and *et al.*, "Performance and reliability of tree-structured grid services considering data dependence and failure correlation," *IEEE Trans. Computers*, vol. 56, pp. 925–936, 2007.

[2] V. Ravi and *et al.*, "Nonequilibrium simulated-annealing algorithm applied to reliability optimization of complex systems," *IEEE Trans. Reliability*, vol. 46, pp. 233–239, 1997.

[3] V. Shestak and *et al.*, "Robust sequential resource allocation in heterogeneous distributed systems with random compute node failures," in *Proc. IPDPS '09*, Rome, Italy, 2009.

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

[5] R. Sheahan and *et al.*, "The effect of different failure recovery procedures on the distribution of task completion times," in *Proc. IEEE DPDNS '05*, Denver, CO, 2005.

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

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

[8] S. Dhakal and *et al.*, "Dynamic load balancing in distributed systems in the presence of delays: A regeneration-theory approach," *IEEE Trans. Parallel and Dist. Systems*, vol. 18, pp. 485–497, 2007.

[9] ——, "A regeneration-based approach for resource allocation in cooperative distributed systems," in *Proc. ICASSP*, 2007, pp. III–1261–III–1264.

[10] S. Dhakal, "Load balancing in communication constrained distributed systems: A probabilistic approach," Ph.D. dissertation, University of New Mexico, 2006.