Optimal Task Reallocation in Heterogeneous Distributed Computing Systems with Age-dependent Delay Statistics

Jorge E. Pezoa*,†, Majeed M. Hayat‡, Zhuoyao Wang* and Sagar Dhakal§

*Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87131, USA
E-Mail: {jpezoa,hayat}@ece.unm.edu, zywang@unm.edu
†J. E. Pezoa is also with Electrical Engineering Department, Universidad de Concepción, Concepción, Chile
‡M. M. Hayat is also with Center for High Technology Materials, University of New Mexico, Albuquerque, NM, USA
§Naval Research Laboratory (NRL), Washington, DC, 20375, USA; E-Mail: sagar.dhakal@gmail.com

Abstract—This paper presents a general framework for optimal task reallocation in heterogeneous distributed-computing systems and offers a rigorous analytical model for the stochastic execution time of a workload. The model takes into account the heterogeneity and stochastic nature of the tasks’ service and transfer times, servers’ failure times, as well as an arbitrary task-reallocation policy. The stochastic service, transfer and failure times are assumed to have general, age-dependent (non-exponential) distributions, resulting in a tandem distributed queuing system with non-Markovian dynamics. Auxiliary age variables are introduced in the analysis to capture the memory associated with the non-Markovian stochastic times, thereby enabling a regenerative age-dependent analytical characterization of the statistics of the execution time of a workload. The model is utilized to devise task reallocation policies that optimize three metrics: the average execution time of a workload, the quality-of-service in executing a workload by a prescribed deadline and the reliability in executing a workload. Implications of the non-exponential event times on these metrics are also studied. Key results are verified experimentally on a distributed-computing testbed.

Index Terms—regeneration time, distributed computing, task reallocation, load balancing, communication delays, non-Markovian queues

I. INTRODUCTION

A distributed computing system (DCS) is a distributed-memory, multi-computer environment that provides its users means to serve large, computationally intensive workloads. Unlike parallel computing environments, server nodes are geographically dispersed and offer heterogeneous computing capabilities. In addition, the communication network interconnecting the servers typically exhibits both low bandwidth and a significant latency in the information exchange.

In order to exploit the processing capabilities of a DCS, workloads are partitioned into small independent tasks. Such tasks are allocated onto the servers for their parallel execution. The problem of efficiently allocating a batch of tasks for their execution in a DCS is a fundamental problem in distributed computing. Solutions to this problem can be obtained by devising run-time control actions, termed as dynamic task reallocation (DTR) policies, that optimize some predefined performance metric, while efficiently using the resources of the DCS [1]–[8]. These DTR strategies strongly rely on (1) an accurate estimate of the state of the DCS and (2) an accurate model for the system. Since the communication network in a DCS imposes non-negligible, stochastic communication delays, accurately estimating the number of tasks allocated at the servers and the number of functioning servers is a non-trivial problem because such estimates must be constructed using possibly dated and/or incomplete information on the servers’ states.

The dynamics of a DCS is complex due to multiple, concurrent stochastic events that govern the behavior of the entire system. Consequently, it is extremely difficult to develop simple analytical models that can accurately predict the dynamics of such systems. In order to circumvent this complexity, a large number of mathematical models for DCSs are based on heuristics, graphical modeling tools like Petri Nets, Monte-Carlo (MC) simulation, and assumptions on the deterministic behavior of the transfer time of tasks [9]–[12]. While the latter assumption is meaningful in parallel computing systems, it has been argued that due to the uncertainty introduced by both the communication network and the number of functioning servers; as such, stochastic models for DCSs must be considered in solving task reallocation problems [13], [14].

The most widely used stochastic model for DCSs is the Markovian distributed queueing network. Such model is obtained under the assumption that all the concurrent events in the system follow exponential distributions. The extensive amount of research on Markovian models has yielded several modeling and analysis tools such as TimeNET and WebSPN [10], [15]–[17]. However, due to physical constraints of real systems, the probability distribution of random times of some events, such as the task service time and the task transfer time, cannot be accurately modeled by an exponential distribution. For instance, in practical communication networks a non-zero end-to-end propagation delay is always observed in any exchange of information. An exponential distribution is clearly

Sagar Dhakal is a postdoctoral fellow at NRL. This work was performed separately and does not represent any view from NRL.
not capable of capturing such minimum transfer delay. Moreover, in [7] our simulations showed that Markovian models for the execution time of a workload are inaccurate in situations where the average task-transfer delays are large compared to the average task service-time of the servers. Furthermore, we also observed that the empirical characterizations for both task service times and task transfer times seem to follow Pareto and shifted Gamma distributions.

In this paper we have extended our earlier characterization of a DCS by relaxing the assumption on the exponential distribution of the random times governing the dynamics of a DCS. To this end, we have introduced in our analysis an auxiliary continuous-time age matrix, which keeps track of the memory of all the non-exponential random times. The age matrix augments the state-space model presented in [2], [7] yielding a hybrid continuous and discrete state-space. This hybrid setting has enabled the development of an age-dependent stochastic regeneration theory, leading to the analytical characterization of the execution time of a workload. The characterization has been, in turn, utilized to calculate three performance metrics, namely, the average execution time of a workload, quality-of-service (QoS) guaranteed in executing a workload, and the service reliability in processing a workload. In order to optimize these metrics, in our age-dependent model for the execution time we have also considered that an arbitrary DTR policy is executed by all the servers. Thus, optimal DTR policies are devised by searching over all the feasible policies using a simple yet effective iterative algorithm. Notably, our policies are devised by searching over all the feasible policies and task transfer times seem to follow Pareto and shifted Gamma distributions.

This paper is organized as follows. In Section II we build the age-dependent regeneration theory for the execution of workloads in a DCS. In Section III we compare the age-dependent approach with our previous Markovian regenerative solution; in particular, DTR policies for minimal average service time, maximal QoS and maximal reliability are devised and compared. Our conclusions are given in Section IV.

II. THEORY

A. Problem statement

Consider the problem of scheduling a workload for its distributed execution on a DCS composed of \( n \) servers. Suppose that such workload can be divided into \( M \) tasks, where a task is an indivisible and independent unit of work that can be executed at any server in the DCS, and suppose also that, at \( t = 0 \), the \( j \)th server has \( m_j \) tasks in its queue, with \( m_j \) a non-negative integer number and \( M = \sum_{j=1}^{n} m_j \). It has been assumed here that the time taken by any server to execute a task is random, and that servers can fail permanently at any random instant. In addition, we have supposed that servers communicate each other through a network that imposes random, non-negligible transfer delays to each message exchanged. We have assumed also that queue-length information messages are frequently exchanged by the servers. The information on these messages is used by the servers to estimate the queue-length of the remaining servers in the system.

This paper aims to devise a DTR policy that optimizes a predetermined performance metric. A DTR policy is a key control mechanism used to efficiently employ the computing resources of the DCS, and simultaneously, compensate for the effects of communication delays and server failures. A DTR policy specifies how many tasks must be reallocated between any pair of nodes in the DCS. Let \( L_{ij} \) denote the number of tasks to be reallocated from the \( i \)th to the \( j \)th server. By arranging the \( L_{ij} \) quantities in matrix form, we can denote a DTR policy by the matrix \( L = (L_{ij})_{n \times n} \). In this paper three performance metrics are considered: the average execution time of a workload, the service reliability and the QoS in executing a workload. The average execution time is critical to speed-up the execution of applications and it is a reasonable metric (i.e., it takes a finite value) only in settings where servers are completely reliable. The service reliability is important in assessing the reliability of DCSs and it is a reasonable performance metric only when servers can fail permanently with non-zero probability. Finally, the QoS metric can be defined in any setting and it is of interest in real-time applications. Mathematical definitions of these metrics are stated later.

The following assumptions are imposed on the random times characterizing the DCS:

**Assumption A1**: For any \( j \neq k \), the following times are random and their probability distribution functions (pdfs) are known: (i) \( W_{ik} \): the service time of the \( i \)th task at the \( k \)th server, with pdf \( f_{W_{ik}}(x) \); (ii) \( Y_{k} \): the failure time of the \( k \)th server, with pdf \( f_{Y_{k}}(x) \); (iii) \( X_{jk} \): the transfer time of a failure-notice (FN) message sent from the \( j \)th to the \( k \)th server, with pdf \( f_{X_{jk}}(x) \); and (iv) \( Z_{ik} \): the transfer time of the \( i \)th group of tasks sent to the \( k \)th server, with pdf \( f_{Z_{ik}}(x) \).

**Assumption A2**: For mathematical tractability, all the random variables listed in Assumption A1 are mutually independent.

Additionally, here it has been assumed that the DCS does not provide any mechanism to recover tasks from a failed server. Also, we have assumed that servers employ a reliable message-passing protocol. With this, tasks cannot be discarded in the network in situations like the failure of a server while exchanging tasks with other nodes. The problem of task reallocation for reliability where fault-tolerance mechanisms are provided by the DCS has been studied in earlier works [7], [18].

B. State-space model for the execution time of a workload

In [2], [7] we presented a discrete state-space model for the random time taken by the DCS in executing a workload under a Markovian setting (i.e., when all the random times are exponentially distributed). For completeness, we review

\footnote{For convenience, the following random times are set to infinity in degenerate cases: (i) the service time at a faulty server or at a server with no task; (ii) the failure time of a faulty server; and (iii) the transfer time of either a FN packet or a group of tasks when no FN packet or group of tasks is in transit.}
here the germane definitions. In a Markovian setting, the configuration of an $n$-server DCS can be described using only three fundamental quantities: (i) the $n$-dimensional, column vector $M$ whose $i$th component specifies the number of tasks queued at the $i$th server; (ii) the $n$-by-$n$, binary matrix $F$ whose $ij$th element describes the functional or dysfunctional state of the $j$th server as perceived by the $i$th server; and (iii) the network-state matrix $C$ that specifies the number of tasks in transit over the network to each server. These state matrices, however, are not sufficient for describing a non-Markovian DCS where random times have memory. One of the main contributions of this paper is the development of an age-dependent extension of the state-space models in [2], [7]. This extension has enabled us to: (i) construct a general representation for the configuration of a DCS, and (ii) rigorously model the random time taken by the DCS in executing a workload in a non-Markovian setting. This is discussed next.

1) Auxiliary age variables: Let $T$ be a non-negative random variable with pdf $f_T(t)$. Intuitively speaking, if we know that $T > a$, then we can think of the aged version of $T$, denoted by $T_a$, as the “replacement” of $T$ measured from $a$. More precisely, the age parameter, $a$, associated with the random variable $T$, is a non-negative, real-valued quantity that defines, on the event $A = \{T \geq a\}$, the random variable $T_a = T - a$. The random variable $T_a$, the aged version of $T$, has a pdf equal to the conditional pdf of $T$ given that $A$ has occurred. Namely, $f_{T_a}(t; a) = f_{T|A}(t|a)$. (Note that if $T$ is an exponentially distributed random variable, then the pdfs of $T$ and $T_a$ are identical due to the memoryless property of the exponential distribution.) Here we exploit the relationship between a random variable and its aged version as follows: as soon as a random time $T$ is triggered by some event, its associated age variable is set to zero, and as time elapses, the age variable keeps track of the age of $T$ and accordingly adjusts the pdf of $T$ to show the effect of the elapsed time on its likelihood. In what follows, we will associate each random time an aged-version of it. This enables us to specify the “age” of the DCS and will further enable us to characterize recursively the execution time of a workload.

Let $a_{M_i}$ and $a_{F_{ij}}$ be age variables associated with the random service time of a task at the $i$th server and the random failure time of the $j$th server, respectively, with $i = 1, \ldots, n$. Also, let $a_{F_{ij}}$ be the age variable connected to the random transfer time of a FN packet from the $i$th to the $j$th server, with $i, j = 1, \ldots, n, i \neq j$. We can arrange all these age variables in the column vector $a_M$ and the $n$-by-$n$ matrix $a_F$. The $a_M$ vector contains the $a_{M_i}$ age variables and the $a_F$ matrix contains both the $a_{F_{ij}}$ variables (at the diagonal of the matrix) and the $a_{F_{ij}}$ variables (at the off-diagonal positions).

Similarly, let $a_{C_{k,t}}$ be age variables associated with the random transfer of the $k$th group of tasks to the $i$th server, with $k$ a positive integer. We can also arrange these age variables in matrix form to obtain $a_C$, whose $ik$th component is $a_{C_{k,t}}$.

We define the system-age as the concatenated matrix $a = (a_M, a_F, a_C)$. Further, for a given time $t$ we define the age-dependent system-state matrix as the concatenated matrix $S(t) = (M(t), F(t), C(t), a(t))$, which describes completely the state of an $n$-server DCS. (Note that in a Markovian setting the memoryless property of the exponential distribution makes the system-age matrix unnecessary; therefore, the system-state matrix reduces to $S(t) = (M(t), F(t), C(t))$ as in [7].) With this we can introduce the stochastic process $\{S(t), t \geq 0\}$ characterizing the stochastic dynamics of the DCS.

With the above necessary preliminaries, we can define and analyze the execution time of a workload and the performance metrics described earlier. The workload execution time is defined as the random time taken by the DCS to serve its entire workload when the initial system configuration is as specified by $S_0 = S(0)$. More precisely, we define $T(S_0) \triangleq \inf\{t > 0 : M(t) = 0 \text{ and } C(t) = 0\}$. Note that since servers can fail permanently with non-zero probability, the workload execution time is infinite when at least one task remains queued at a server that has already failed. Note also that in the case where servers are completely reliable, the workload execution time is always finite.

The average execution time of a workload, denoted as $\bar{T}(S_0)$, is defined as the expected value of the random workload execution time. Mathematically, we have $\bar{T}(S_0) \triangleq E[T(S_0)]$. This metric is defined in the case of a DCS with completely reliable servers; that is, when $Y_k = \infty$ almost surely for all $k$. The service reliability is defined as the probability that the workload can be entirely executed by the system, that is $R_\infty(S_0) \triangleq P\{T(S_0) < \infty\}$. Finally, the QoS in executing a workload is the probability that the workload can be entirely executed by the time $T_M$; that is $R_{T_M}(S_0) \triangleq P\{T(S_0) < T_M\}$. Note that the service reliability is a special case of QoS for which the time constraint to execute the workload is finite.

Finally, recall that at $t = 0$ an arbitrarily specified DTR policy, $L$, is executed by all the servers; therefore, the metrics defined here are also a function of $L$. In Section II-C2, Theorem 1 and Remark 1, we recursively characterize the average execution time and the QoS in executing a workload in a DCS.

C. Age-dependent regeneration-based approach and recursive characterization of the performance metrics

1) Rationale: We define an age-dependent regeneration event and analyze the stochastic process emerging immediately after the first occurrence of this event. The age-dependent regeneration event is defined as the first occurrence of either the service of a task at any server, the failure of any server, the reception of a FN packet by any server, or the reception of a group of tasks by any server. The point here is that upon the occurrence of the regeneration event, a fresh copy of the original stochastic process emerges at the regeneration time albeit with a new initial configuration that transpires from the
regeneration event. In a Markovian setting, the memoryless property of the exponential distribution guarantees that the process regenerates itself at any time. However, when the stochastic process is not Markovian, it is necessary to keep track of the memory of all non-exponential distributions in order to yield a regenerative process. Therefore, auxiliary age variables, \( a_i \), introduced in the previous section must be included in the description of the process configuration.

Recall the process \( \{S(t), t \geq 0\} \) and suppose that at time \( t = 0 \) the system configuration is as specified by \( S_0 = (M_0, F_0, C_0, a_0) \). Formally, we define the age-dependent regeneration time, denoted by \( \tau_a \), as the minimum of the following four random variables: the time to the first task service by any server, the time to the first occurrence of failure at any server, the time to the first arrival of a FN packet at any server, or the time to the first arrival of a group of tasks at any server. Mathematically, \( \tau_a \triangleq \min(\min_i W_{k_i}, \min_k Y_k, \min_j \neq k Y_{jk}, \min_i \xi_i, Z_{ik}) \). The upcoming example describes how the age-dependent regeneration time and system-state matrix yield a recursive characterization for the workload execution time.

Suppose that the first event occurring in the DCS happens to be the execution of a task at the \( i \)th server at \( t = s \). The occurrence of this event implies that all the random times governing the DCS have aged by \( s \) units of time and there is one less task queued at the \( i \)th server; all the other dynamics remain unchanged. Thus, the occurrence of the event \( \{\tau_a = s, \tau_a = W_{i1}\} \) gives birth to a new DCS at \( t = s \), represented by \( \{S(t), t \geq s\} \), that is statistically identical to the original process while having a new initial configuration, \( S(t) = (M_0, F_0, C_0, a_0) \), resulting from the regeneration event \( \{\tau_a = s, \tau_a = W_{i1}\} \).

More precisely, the new initial system configuration is as follows: \( M_0 = M_0' \) but with one unit less at its \( i \)th element, \( F_0 = F_0', C_0' = C_0 \), and the new system-age matrix is \( a_0' = a_0 + s \) with the \( i \)th component of \( a_M \) set to zero if at least one task remains queued at the \( i \)th server. Similar transformations on the initial configuration are observed when the regeneration event is any of the remaining events, namely, the failure of the \( i \)th server, the arrival of a FN packet from server \( j \) to server \( k \), or the arrival of the \( i \)th group of tasks to the \( k \)th server.

2) Characterization of the performance metrics: Before stating Theorem 1, we introduce some useful definitions. Let us define the term \( G_X(s) \triangleq \mathbb{P}(X = \tau_a | \tau_a = s) f_{\tau_a}(s) \), where \( X \) is any of the random times listed in Assumption 1. \( f_{\tau_a}(s) \) is the pdf of the age-dependent regeneration time \( \tau_a \) and \( \mathbb{P}(X = \tau_a | \tau_a = s) \) is the probability that the regeneration event is \( \{\tau_a = X\} \) conditional on the event \( \{\tau_a = s\} \). This conditional probability can be computed explicitly, either analytically or numerically, using Assumptions 1 and 2. For ease of the presentation we show here the two-server case. Let \( M = (m_1, m_2) \), \( F = (f_{j1}, f_{j2}) \), and \( C = (g_{11}, L_{11}, ..., L_{i1}) \), with \( f_{ij} \in \{0, 1\} \) for \( i, j = 1, 2 \), \( g_{il} \) is a non-negative integer representing the number of groups of tasks in transit to the \( i \)th server, and \( L_{kj} \) is the number of tasks in the \( k \)th group in transit to the \( j \)th server, \( k \in \{1, \ldots, g_i\} \). This characterization can be extended to an \( n \)-server system in a straightforward manner.

Theorem 1, whose sketch of proof is given in the Appendix, characterizes the average workload execution time and the QoS of a DCS.

**Theorem 1.** Consider a two-server DCS with an arbitrarily specified initial system configuration \( S_0 = (M, F, C, a_M, a_F, a_C) \). The average workload execution time satisfies:

\[
T(M, F, C, a_M, a_F, a_C) = E[\tau_a] + \int_0^\infty \left[ \sum_{i=1}^{2} G_{W_i}(s) \right] ds,
\]

where the QoS in executing a workload satisfies:

\[
R_{T_M}(M, F, C, a_M, a_F, a_C) = \int_0^\infty \left[ \sum_{i=1}^{2} G_{W_i}(s) \right] ds,
\]

\[
R_{T_M-s}(M, F, C, a_M+s, a_F, a_C+s) = \int_0^\infty \left[ \sum_{i=1}^{2} G_{W_i}(s) \right] ds,
\]

\[
R_{T_M-s}(M, F, C, a_M+s, a_F, a_C+s) = \int_0^\infty \left[ \sum_{i=1}^{2} G_{W_i}(s) \right] ds,
\]

\[
R_{T_M-s}(M, F, C, a_M+s, a_F, a_C+s) = \int_0^\infty \left[ \sum_{i=1}^{2} G_{W_i}(s) \right] ds,
\]

\[
R_{T_M-s}(M, F, C, a_M+s, a_F, a_C+s) = \int_0^\infty \left[ \sum_{i=1}^{2} G_{W_i}(s) \right] ds,
\]

where \( \delta_{ij} \) is the Kronecker delta, \( m_i = 1 \) is set to zero when \( m_i = 0 \), and the vector \( v^{(i)} \) (correspondingly, matrix \( A^{(ij)} \)) is identical to \( v \) (correspondingly, \( A \)) but with its \( i \)th (correspondingly, \( ij \)) component set to zero.

**Remark 1.** Non-Markovian representations for the metrics in Theorem 1 in the case of an \( n \)-server DCS can be obtained in a straightforward manner following the same principles as those for a two-server system.
In order to compute, say, the service reliability we must consider the system configuration at \( t = 0 \). This configuration is: (i) \( M_0 = (r_2) \) where \( r_1 \) is the number of tasks queued at the \( i \)th server, with \( r_i = m_i - L_{ij} \), \( i = 1, 2, i \neq j \). \( L_{ij} \) is the number of tasks reallocated from the \( i \)th to the \( j \)th server according to an arbitrary DTR policy executed at \( t = 0 \); (ii) \( F_0 \) is an all-ones matrix because at \( t = 0 \) both servers are assumed to be functioning; (iii) \( L_{12} \) and \( L_{21} \) tasks are being transferred in the network; therefore, \( g_1 = g_2 = 1 \), \( L_{1,1} = L_{2,1} \) and \( L_{1,2} = L_{12} \); and (iv) the system-age matrix is the null matrix. After plugging \( S_0 \) in (2), we obtain a recursion for \( R_\infty(S_0) \) in terms of \( r_1 \) and \( r_2 \), the number of tasks queued at the servers after executing an arbitrary DTR policy.

It turns out that to solve this recursion not only the values of \( S_0 \) for \( r_1 - 1 \) and \( r_2 - 1 \) 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 the servers and when no tasks are in transit in the network. Consequently, starting with \( S_0 \) and (2) we have to construct a system of recurrence equations, which has to be solved following a particular order. Equations forming such system are derived in a straightforward manner (details not shown for brevity) using (2) and the new initial configurations shown at the right hand side of (2). Finally, recursions are solved using the following initial conditions: \( R_\infty(S_0) = 1 \) when there are no tasks to be served in the DCS and \( R_\infty(S_0) = 0 \) when at least one server has failed and tasks remain unserved in its queue.

**Differences between the Markovian and the non-Markovian models:** At this point we highlight four differences between the Markovian and non-Markovian characterizations for the metrics regarded here. First, in the Markovian setting the conditional probabilities associated with the regeneration event remain constant, while in the non-Markovian case are age dependent. Second, in the Markovian setting the regeneration time follows an exponential distribution, while in the non-Markovian case the regeneration time follows a general distribution whose parameters are age-dependent. Third, the state-space representation for the workload execution time is discrete in the Markovian case, while in the non-Markovian case is a hybrid of discrete and continuous representation. Fourth, in the Markovian setting the three performance metrics are characterized by algebraic recurrence equations with constant coefficients, while in the non-Markovian scenario recursions comprise integrals with age-dependent coefficients.

**D. Policy for optimal task reallocation: 2-server case**

Recall that the random execution time is also a function of the DTR policy \( L \) executed by all servers at \( t = 0 \), that is \( T(S_0) \equiv T(L; S_0) \). For a two-server system, we can employ the models given in Theorem 1, with the initial system configuration \( S_0 \), to search for the DTR policy, \( L^* = (L^*_{12}, L^*_{21}) \), that optimizes any of these three performance metrics. Formally, we have the optimization problems:

\[
(L_{12}, L_{21}^*) = \arg\min_{(L_{12}, L_{21})} T(L; S_0), \tag{3}
\]

\[
(L_{12}^*, L_{21}^*) = \arg\max_{(L_{12}, L_{21})} R_{T_m}(L; S_0), \tag{4}
\]

Both subject to: (i) \( L_{ij} + r_i = m_i \) for \( i = 1, 2, i \neq j \); and (ii) \( L_{ij} \) an integer number in \([0, m_i]\), for \( i = 1, 2, i \neq j \).

**Scalability issues:** For DCSs with arbitrary number of servers, we can attempt to solve the optimization problem using the \( n \)-server characterization for the performance metrics; however, computing the metrics using the exact \( n \)-server characterization is expensive as the number of computations grows exponentially in the number of servers. As an alternative, for \( n \)-server systems we follow [7] and provide a suboptimal algorithm for DTR policies that scales linearly with the number of servers. The key idea is to decompose an \( n \)-server system into several two-server DCSs and exploit the exact characterization for two-node systems.

**E. Scalable policy for approximately optimal task reallocation: multi-server case**

The algorithm computes the number of tasks to reallocate from the \( i \)th to the \( j \)th server at the \( k \)th iteration, \( L_{ij}^{(k)} \), as follows. The \( i \)th server has an estimate, \( \hat{m}_{ij}, \) of the number of tasks queued at the \( j \)th server. Using these estimates, the \( i \)th server constructs its collection of candidate task-recipient servers, \( U_i \). From such collection, the \( i \)th server picks the \( j \)th server, say, and obtains \( L_{ij}^{(k)} \) by solving either (3) or (4) with \( m_1 = r_i \) and \( m_2 = \hat{m}_{ij} \), where \( r_i \) is the number of tasks queued at the \( i \)th server assuming that such server has already reallocated tasks to all its other candidate recipient servers, with the exception of the \( j \)th server. In order to produce an algorithm independent of the order in which servers are selected from \( U_i \), the \( L_{ij}^{(k)} \) quantities are iteratively computed until all of them converge to some value or until a maximum number of iterations, \( K \), is reached. A pseudocode for the algorithm is shown in Algorithm 1. Note that each server has to solve at most \( n-1 \) times any of the 2-server optimization problems, and such computation has to be repeated no more than \( K \) times. From this, we observe that the complexity of the algorithm increases linearly in the number of servers.

The algorithm requires the following parameters: \( K \), \( \hat{m}_{ij} \), and \( L_{ij}^{(0)} \). The parameter \( K \) is selected by the user. The estimates \( \hat{m}_{ij} \) are obtained from queue-length information packets frequently exchanged among the servers. Finally, the initial DTR policy, \( L_{ij}^{(0)} \), is computed using:

\[
L_{ij}^{(0)} = \left\lfloor \hat{m}_{ij} - \hat{M}_i \frac{\sum_{\ell=1}^{n} A_{ij}^{-1} x_{ij}}{\sum_{\ell=1}^{n} A_{ij}^{-1}} \right\rfloor, \tag{5}
\]

where \( \lfloor x \rfloor \) is the greatest integer smaller than or equal to \( x \), \( \hat{M}_i = m_i + \sum_{j=1, j \neq i}^{n} \hat{m}_{ij} \) is the total load in the system (as estimated by the \( i \)th server), \( A_{ij} \) is a parameter that can be defined in several ways in order to establish different reallocation criteria. For example, if \( A_{ij} \) is equal to the processing speed of the \( j \)th server, then the reallocation criterion is determined by the relative computing power of the servers. Alternatively, if
Algorithm 1 DTR policy for multi-server DCSs

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

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

loop
  while $j \in U_i$ do
    $U_i \leftarrow U_i \setminus \{j\}$
    $m_1 = m_1 - \sum_{k \in U_i} L_{ij}^{(k-1)} - \sum_{k \in U_i'} L_{ij}^{(k)}$ and $m_2 = \hat{n}_{j,i}$
    Solve (3) or (4) using $m_1$ and $m_2$ to obtain $L_{ij}^{(k)}$
    $U_i' \leftarrow U_i' \cup \{j\}$
  end while

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

if $\sum_{j=1}^{n}(L_{ij}^{(k)} - L_{ij}^{(k-1)}) > 0$ or $k > K$ then
  $L_{ij}' = L_{ij}^{(k)}$ for all $j \in U_i$ and exit
end if

end loop

$\Lambda_j$ is equal to the reliability of the $j$th server, then the relative server reliability defines the reallocation criterion.

III. RESULTS

A. Comparing Markovian and non-Markovian models

Let us compare predictions for the average execution time, the service reliability and the QoS in executing a workload obtained using the non-Markovian characterization to those provided by the Markovian models in [2], [7]. In our evaluations the communication network is assumed to be homogeneous and two network-delay conditions have been considered: low and severe network-delays. Under low network-delay conditions, transferring a task to and processing such task at the fastest server takes, on average, the same time as processing the task at the slowest server. For the severe network-delay case, the average transferring plus processing time of a task to the fastest server is at least five times the service time at the slowest server.

We have employed different stochastic models for the service and transfer times. The Markovian setting is represented by the Exponential model. Pareto 1 and Pareto 2 models represent the case where service and transfer times follow Pareto laws with finite and infinite variance, respectively. For the Shifted-Exponential model, both service and transfer times follow shifted exponential distributions. Finally, in the Uniform model service and transfer times follow uniform distributions. For fair comparison, all distributions modeling the same random times have identical means.

1) Task reallocation for two-server systems: Markovian and non-Markovian models are first compared in a heterogeneous DCSs composed of two servers. The workload supplied to the system comprises $m_1 = 100$ tasks and $m_2 = 50$ tasks, initially allocated at servers 1 and 2, respectively. The mean service time per task is 2 and 1 s for servers 1 and 2, respectively. We have also assumed that failure times follow exponential distributions with means $\lambda_{f1}^{-1} = 1000$ and $\lambda_{f2}^{-1} = 500$ s. (Recall that when the average execution time is calculated, servers are assumed to be 100% reliable.) Also, the mean transfer time of FN packets are 0.2 and 1.0 s for the low and severe delay scenarios, respectively.

Figures 1 and 2 show the average execution time and the service reliability as a function of several DTR policies, and for the two network-delay conditions considered. In these DTR policies the number of task reallocated from server 2 to 1, $L_{21}$, is 25 tasks (50% of the initial load is reallocated from the faster to the slower server). It can be noted that the Markovian approximations for both metrics, the average execution time and the service reliability, show a remarkable accuracy in the low network-delay condition. In fact, the maximum relative approximation errors are below 3% in all cases for both metrics. However, as the mean transfer time increases, Figs. 1 and 2 also show that the Markovian approximations loose their accuracy in predicting the actual values of the metrics. When network-delays are severe, maximum relative approximation errors increase up to 15% for the average execution time and up to 65% in the case of the service reliability.

Figures 1 and 2 also show the effect of the network-delays on the performance metrics. It can be noted that as the mean
transferring a workload from server 1 to server 2 has to be reduced as compared to the case of low network delays. It can also be noted that policies aiming to reduce the execution time of a workload are not appropriate for maximizing the service reliability. Interestingly, the optimal policy for reliability does not achieve the minimal average execution time of a workload. This situation is due to policies minimizing execution time exploiting the processing capability of the faster server, and such requirement conflicts with the needs of policies aiming for maximizing service reliability, which aim to utilize the most reliable yet slower server a longer time. A trade-off between minimizing execution time and maximizing service reliability can be obtained by devising policies that simultaneously optimize the two performance metrics.

We solve now the optimization problems (3) and (4) to devise DTR policies for the average execution time and the QoS in executing a workload. For both performance metrics we have considered the case of a DCS with completely reliable servers. Optimization results for all the models and both low and severe network-delay conditions are listed in Table I. It can be seen from such Table that when network-delays are low, a Markovian model yields fairly accurate predictions for both metrics. However, when the amount of delay in the communication network is high, results produced by the Markovian approximation are not only inaccurate, but also yield DTR policies that can degrade the performance metrics in approximately 10 to 40%. Figure 3 shows the average execution time and the QoS in executing a workload as a function of the DTR policies for the Pareto 1 model and severe network-delays. The minimal average execution time is 140.11 s, and is achieved by the policy $L_{12} = 32$ tasks and $L_{21} = 1$ task. Figure 3(b) shows the QoS in executing the entire workload within 180 s. Such metric is maximized by three policies $L_{12} \in \{31, 32, 33\}$ tasks and $L_{21} = 1$ task, which yield a maximal probability of executing the workload of 0.988. We comment that the maximal QoS in executing the workload within 140 s (the minimal average completion time)
is only 0.471.

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

2) Task reallocation for multi-server systems: We have also maximized the average execution time and the service reliability of a heterogeneous, five-server DCS employing the algorithm presented in Section II-E. We have assumed that the workload comprises $M = 200$ tasks. To evaluate the service reliability, we assume that failure times follow exponential distributions with means 1000, 800, 600, 500, and 400 s, for servers 1 to 5, respectively. The average service times were set to be 5, 4, 3, 2, and 1 s for servers 1 to 5, respectively. The remaining parameters are the same as those in the two-server analysis.

Table II lists both the average execution time and the maximal service reliability obtained under severe network-delay conditions. Both performance metrics were obtained through simulations and the values listed in Table II correspond to centers of 95% confidence intervals. For comparison, the column “Exponential” presents results yielded using the optimal policies devised under Markovian assumptions. Also for comparison, the last row on each part of the Table represents a benchmark for each performance metric, since the initial allocation of tasks is actually the optimal allocation. These optimal task allocations were obtained by performing a MC-based exhaustive search over all the DTR policies. It can be noted from Table II that the exponential approximation produces relative errors between 5% and 45%. As in the case of two-server DCSs, using incorrect models for the random times yield not only inaccurate results, but also specifies inappropriate reallocation policies that, in turn, reduce the performance metrics under study. Finally, in our evaluations we have observed that policies devised using Algorithm 1 and a non-Markovian model can achieve values for the average execution times and the service reliability that are within 70% of the optimal values.

**B. Maximizing the service reliability of a testbed DCS**

In order to experimentally validate our theory we devise policies for maximizing the service reliability of a testbed DCS, which uses the Internet as the communication network. A detailed description of the testbed is given in [7]. In order to yield predictions for the service reliability, first we need to characterize experimentally the random times of our testbed DCS. Figures 4(a) and (b) show the normalized histograms as well as fitted pdfs for the service time of server 1 and the transfer time of tasks from server 2 to 1. The parameters of the fitted pdfs were estimated using maximum likelihood estimators. Each estimated pdf was selected according to the minimum total squared error between the normalized histogram and each fitted pdfs. From the experimental characterization, we have that: (i) the service times at servers 1 and 2 follow Pareto distributions with means 4.858 and 2.357 s, respectively; (ii) the task transfer times follow shifted gamma distributions with means $\bar{X}_{12} = 1.207$ and $\bar{X}_{21} = 0.803$ s; and (iii) the FN packet transfer times follow shifted gamma distributions with means $\bar{X}_{12} = 0.313$ and $\bar{X}_{21} = 0.145$ s. Note that according to our classification of delays, these values correspond to a low network-delay condition. The free parameters of the system are the initial workload and the distribution of the failure times of the servers. The initial workload was set to $m_1 = 50$ and $m_2 = 25$ tasks and failure times were assumed to follow exponential distributions with means 300 and 150 s.

The optimal DTR policy calculated using Algorithm 1 and a non-Markovian model for the two-server testbed is $L_{12} = 26$ and $L_{21} = 0$ tasks. Such policy provides a theoretical service reliability of 0.6007. Figure 4(c) shows theoretical predictions, MC simulations as well as experimental results.
for the service reliability of the two-server testbed. Results show the case when the optimal reallocation from server 2 to 1 is used ($L_{21} = 0$), while different number of tasks are migrated from the first to the second server ($L_{12}$). In both simulations and experiments, the service reliability is calculated by averaging failure or success outcomes. A total of 10000 and 500 independent realizations of each policy have been employed in computing MC simulations and experimental results, respectively. Figure 4(c) shows a remarkable agreement between simulations and the non-Markovian theoretical predictions. Experimental results show also a fairly good agreement with the theoretical curves, where the relative error between predictions and experiments is less than 7%. Note that if no task reallocation is performed, the service reliability is reduced in approximately 15%. If a Markovian approximation is employed to devise the optimal DTR policy for the two-server testbed, the service reliability is reduced in approximately 1.5%.

IV. CONCLUSIONS

We have presented a novel analytical characterization for the average execution time of a workload, the service reliability and the QoS in executing a workload in DCSs in the presence of age-dependent communication delays. The age-dependent analytical characterization of these three metrics constitute a generalizations of our earlier Markovian models for the tandem distributed queuing system, reported in [2], [7], to a non-Markovian and hence more physical setting. The analysis of these metrics in the new setting is importance since they provide insight on the effect of network-delays on the accuracy of the Markovian models. Our results indicate that when the network delays are relatively large compared to service times, the error in estimating any of these metrics, as a result of falsely assuming exponentially distributed random delays, becomes significant, thereby necessitating the use of our age-dependent model. For example, our calculations show relative errors as large as 40% and 110% in estimating the average execution time and the service reliability, respectively. It must be mentioned that the accuracy provided by the age-dependent model in predicting the three performance metrics regarded here comes at expense of increased computations as compared to their Markovian counterpart.

In order to capture the memory associated with the non-exponential random times, we have introduced in our model a fundamental quantity, termed as the continuous-time age matrix. This matrix enables us to characterize recursively and analytically the three performance metrics regarded in this paper, while assuming a very general framework. Based on a two-server characterization for such metrics, we presented a scalable algorithm for devising DTR policies that minimize the average execution time of a workload and maximize the reliability and QoS in executing workloads in multi-server DCSs.

Future work will consider to compute approximations as well as bounds for the random workload execution time. Due to the fact that tasks arrive at any random time to the servers, the analysis must consider all possible orders of task-arrival to yield an exact characterization for the DCS. From this, approximations and bounds can be obtained by assuming, for example, that all the tasks reallocated to a server arrive at random, but as a single batch of tasks.

ACKNOWLEDGMENT

This work was supported by the Defense Threat Reduction Agency (Combating WMD Basic Research Program).
We obtain (1) and (2) after applying observations (i)–(iii) to all possible, disjoint regeneration events. That is,
\[
E[T(S)|a_\tau = s] = \sum_{j=1}^{2} P\{a_\tau = W_k|a_\tau = s\} \times E[T(S)|a_\tau = s, a_\tau = Z_{j1}] \tag{9}
\]
\[
P\{T(S) < T_M|a_\tau = s\} = \sum_{k=1}^{2} P\{a_\tau = W_k|a_\tau = s\} \times P\{T(S) < T_M|a_\tau = s, a_\tau = Z_{j1}\} + \sum_{k=1}^{2} P\{a_\tau = Y_k|a_\tau = s\} \times P\{T(S) < T_M|a_\tau = s, a_\tau = Z_{j2}\} \tag{10}
\]
It can be shown that upon the occurrence of a regeneration event, the random times emerging at the regeneration time satisfy Assumptions A1 and A2; that is, the replacement of the original random times, now measured from the regeneration time, satisfy Assumptions A1 and A2. As a consequence, the following observations hold: (i) a fresh copy of the underlying stochastic process emerges at \(a_\tau\) but with a new initial age-dependent system-state matrix; (ii) the emergent stochastic process is independent of the original process and satisfies assumptions A1 and A2; and (iii) the independence of the new process allows us to shift the time origin to \(a\). For example, if the regeneration event is the service of a task at the first server, after applying these observations we can show
\[
E[T(S)|a_\tau = s, a_\tau = W_{11}] = E[T(S)|a_\tau = s, a_\tau = W_{11}] = P\{a_\tau + T(S') < T_M|a_\tau = s, a_\tau = W_{11}\} = \frac{P\{a_\tau + T(S') < T_M|a_\tau = s, a_\tau = W_{11}\}}{r_{T_M}(s)}
\]
where \(T(S')\) is the random time taken by the DCS emerging at the regeneration time to serve all its tasks when the initial configuration is \(S' = (M', F', C, a_M', a_F', a_C')\). This new initial configuration is precisely \(M' = \left( m_{i-1} \right)\). \(F' = F, C' = C, a_M' = \left( a_M + s \right), a_F' = a_F + s,\) and \(a_C' = a_C + s\). By exploiting the independence between the emergent and the original process, and recalling that \(a_\tau = s\), we obtain
\[
E[T(S)|a_\tau = s, a_\tau = W_{11}] = E[S|a_\tau = s, a_\tau = W_{11}] + T(S') = \frac{P\{a_\tau + T(S') < T_M|a_\tau = s, a_\tau = W_{11}\}}{r_{T_M}(s)}
\]
We obtain (1) and (2) after applying observations (i)–(iii) to all the remaining regeneration events and recombining terms.