SIAM J. COMPUT. © 2005 Society for Industrial and Applied Mathematics
Vol. 34, No. 3, pp. 616–639

LOAD BALANCING IN ARBITRARY NETWORK TOPOLOGIES WITH STOCHASTIC ADVERSARIAL INPUT[*]

ARIS ANAGNOSTOPOULOS[†], ADAM KIRSCH[‡], AND ELI UPFAL[†]

Abstract. We study the long-term (steady state) performance of a simple, randomized, local load balancing technique under a broad range of input conditions. We assume a system of n processors connected by an arbitrary network topology. Jobs are placed in the processors by a deterministic or randomized adversary. The adversary knows the current and past load distribution in the network and can use this information to place the new tasks in the processors. A node can execute one job per step, and can also participate in one load balancing operation in which it can move tasks to a direct neighbor in the network. In the protocol we analyze here, a node equalizes its load with a random neighbor in the graph.

Our analysis of the protocol does not assume any particular input distribution. The input is generated by an arbitrary deterministic or probabilistic adversary subject only to some weak statistical properties. For stability and expected performance of the system we adopt the stochastic adversary model of [Borodin et al., *J. ACM*, 48 (2001), pp. 13–38]. For high-probability bounds we introduce a more restricted input model, the *strongly bounded* adversary.

Assuming the stochastic adversarial input model, we show that if the adversary does not trivially overload the network (i.e., there is an integer $w \geq 1$ such that the expected number of new jobs in any interval of length w is bounded by $\lambda n w$ for some $\lambda < 1$), then the system is stable for any connected network topology, regardless of how the adversary allocates the new jobs between the processors.

When the system is stable, the next performance parameter of interest is the waiting time of jobs. We develop expected and high probability bounds on the total load in the system and the waiting time of jobs in terms of the network topology. In particular, in the above stochastic adversary model, if the network is an expander graph, the expected wait of a task is $O(w + \log n)$, and in the strongly bounded adversary model the waiting time of a task is $O(w + \log n)$ with high probability.

We contrast these results with the work stealing load balancing protocol, where we show that in sparse networks, the load in the system and the waiting time can be exponential in the network size.

Key words. dynamic load balancing, stochastic adversary, stability, efficiency, steady state analysis

AMS subject classifications. 68W40, 68W20, 60G35

DOI. 10.1137/S0097539703437831

1. Introduction.

Efficient utilization of parallel and distributed systems can often depend on dynamic load balancing of individual tasks between processors. In the dynamic load balancing problem, we consider a system that is designed to run indefinitely. New jobs arrive during the run of the system, and existing jobs are executed by the processors and leave the system. The arrival of new jobs may not be evenly distributed between the processors. The task of the load balancing protocol is to maintain approximately uniform job load between the processors, and in particular to keep all processors working as long as there are jobs in the system waiting for execution.

We assume a simple combinatorial model of load balancing following a number of earlier studies. The computing system is represented by a connected, undirected,

[*]Received by the editors November 20, 2003; accepted for publication (in revised form) August 20, 2004; published electronically March 17, 2005.
http://www.siam.org/journals/sicomp/34-3/43783.html

[†]Department of Computer Science, Brown University, Providence, RI 02912 (aris@cs.brown.edu, eli@cs.brown.edu).

[‡]Division of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138 (kirsch @eecs.harvard.edu). Part of this work was done while the author was at Brown University, and was supported in part by NSF grants CCR-0121154 and DMI-0121495.

n-node graph. Jobs (represented by tokens) have equal execution time. The load of a node is the number of tokens in its queue. A processor can execute (or consume) one token per step, and we assume that it executes the oldest job in its queue. In each step a processor can also move a number of jobs from its queue to the queue of an adjacent node in the network. This abstraction models the case where the execution time of a job is significantly longer than the time required to move a job to an adjacent node. The assumption that all jobs have equal execution time simplifies the analysis while still capturing the combinatorial complexity of the load balancing problem in networks.

Dynamic load balancing algorithms have been studied extensively in experimental settings, demonstrating significant run-time improvements obtained by relatively simple load balancing techniques [20, 21]. Rigorous, theoretical study of load balancing in the past has focused mainly on static analysis [1, 7, 8, 11, 16, 18], where a set of jobs is initially placed in the processors and the algorithm needs to distribute the jobs almost evenly between the processors in a minimum number of parallel rounds. A number of important techniques have been developed in this line of work, and in particular our work here builds on the static analysis in [8].

Load balancing, however, is best analyzed in a dynamic setting that captures the actual application of such protocols. An important step in that direction was taken in [4], where the work stealing model was shown to be stable on the complete network. The main tool used in that work was the stability conditions for ergodic Markov chains, and consequently their stability result holds only for Markovian adversaries. In addition, a number of other works have studied dynamic load balancing under the assumptions that jobs are generated by a randomized process that is oblivious to the current state of the system [10, 13, 14, 19]. Finally, [3, 15] proved stability results for load balancing on a general network assuming the deterministic adversarial model. The computational model there is different, assuming that only one job can traverse an edge per step.

2. Model and main results.

In this work we address both the stability and the efficiency (waiting time) of the load balancing task. We present a simple local randomized protocol and analyze its performance on a general n-node network, and under several adversarial models for the arrival of jobs into the system. Since we do not use Markov chain techniques in the analysis, our results are not restricted to Markovian adversaries. That is, the process that injects new jobs into the system can use information about all previous steps.

In particular, we consider the type of adversaries proposed by Borodin et al. in [5]. Following that work we define a $(w, \lambda n)$ input adversary as a process that inserts jobs in the system subject to the condition that for every sequence of w consecutive time steps, the total inserted load is at most $\lambda n w$. This allows the adversary to insert more jobs at some time steps, as long as the total load in windows of size w is bounded. An extension is a $(w, \lambda n)$ stochastic adversary, whose input load is a random variable, with the property that the expected injected load during any sequence of w consecutive time steps is bounded by $\lambda n w$, and additionally, for some $p > 2$, the pth moment of the new load is bounded (see [5] for a detailed discussion).

For these adversaries, we derive asymptotic (with respect to the network size) bounds on both the expected total load and the waiting time of a job in the system. However, it is still reasonable to seek stronger guarantees than bounds on expected performance can provide. In particular, one might wish to augment the expected performance results with high-probability bounds. In this case, the stochastic adversarial model is too powerful, since it allows for large bursts of load to occur in certain time steps with decent probability. Therefore we need to place additional restrictions on the adversary if we wish to derive high-probability results. To this end, we introduce a constrained version of the stochastic adversary by enforcing a large-deviation–type bound for the incoming load, similar to a Chernoff bound. We call these adversaries *strongly bounded*.

Formally, let $\mathcal{A}$ be an adversary that is injecting load into the system. Let I_t be the load injected by the adversary during time step t.

Definition 2.1. *We say that $\mathcal{A}$ is a (λ, w, p, M)* stochastic adversary *(where we assume that w is a positive integer) if the following conditions hold for any time t and any event $\mathcal{H}$ determined entirely by information about the system at or before time t.*

1. $\mathbf{E}\left[\sum_{i=1}^{w} I_{t+i} \,|\, \mathcal{H}\right] \le \lambda n w$.
2. $\mathbf{E}\left[\left(\sum_{i=1}^{w} I_{t+i}\right)^p \,|\, \mathcal{H}\right] \le M n^p w^p$.

In addition, we say that $\mathcal{A}$ is strongly bounded *if it also satisfies the following condition.*

3. *There is a constant $\alpha > 0$ and a constant $\beta \ge 1$ such that for any $\epsilon > 0$*

$$\mathbf{Pr}\left(\sum_{i=1}^{w} I_{t+i} > (1+\epsilon)\lambda n w \,|\, \mathcal{H}\right) \le e^{-\alpha\lambda n w \epsilon^{\beta}}.$$

Throughout the paper, we always assume that λ and p do not depend on n, while w and M may be functions of n.

We give a high-level description of the protocol here, deferring the details to section 3. After the generation of new load, the nodes execute a particular distributed randomized algorithm for choosing a random matching. The matching is not necessarily perfect, nor is it necessarily chosen uniformly from all possible matchings, although it does have some important properties that we will exploit later. Once the matching is chosen, every two matched nodes equalize their load (up to one token). For simplicity, we refer to this protocol as $\mathcal{P}$.

We first show that the system (using $\mathcal{P}$) is stable under the stochastic adversary model (for $\lambda < 1$ and $p > 2$). That is, the expected total load in the system is bounded with respect to time. The following theorem, proven in sections 5.1 and 5.3, relates the load in the system to the network topology and establishes the stability of the system. We assume a connected network $G = (V, E)$ that has n nodes, maximum degree at most Δ, and whose Laplacian[1] has smallest nontrivial eigenvalue Λ. For convenience, we define the quantity $\gamma = \Lambda/16\Delta$.

Theorem 2.2. *Suppose that we run the system with a (λ, w, p, M) adversary, where $\lambda < 1$ and $p > 2$, using protocol $\mathcal{P}$ (described in section 3). Let L_t be the load of the system at time t. Then the system is stable and*

$$\limsup_{t\to\infty} \mathbf{E}[L_t \,|\, L_0] = O(\gamma^{-1} n (w + \ln n)(1+M)^{3p}) \qquad \textit{as } n \to \infty.$$

In addition, if the adversary is strongly bounded, then for any constant $c > 0$ there is a constant $\kappa = \kappa(c)$, such that for sufficiently large n,

$$\liminf_{t\to\infty} \mathbf{Pr}(L_t \le \kappa\gamma^{-1} n (w + \ln n)) \ge 1 - n^{-c}.$$

The above bounds hold without the limits if the system starts with no load.

[1]Let A denote the adjacency matrix of a graph G, and let $D = (d_{ij})$, where d_{ij} is the degree of node i if $i = j$, and is 0 otherwise. The Laplacian of G is the matrix $L = D - A$. The eigenvalues of L are $0 = \Lambda_1 \le \Lambda_2 \le \cdots \le \Lambda_n$, and $\Lambda_2 = \Omega(n^{-2})$ if G is connected.

Next we address the efficiency of the load balancing protocol, an important performance parameter that was not addressed in most previous work. That is, we relate the waiting time of jobs to the topology of the network. Since protocol $\mathcal{P}$ treats all tokens equally regardless of their ages, it cannot guarantee efficient delivery time for individual packets. To bound the waiting time of jobs in the system, we augment the protocol with a distributed version of the *first-in-first-out* queueing discipline, requiring that a node always respect the ages of its tokens in the load balancing step, and always consume its oldest token in the load consumption step (see section 3 for details). We denote this version of $\mathcal{P}$ by $\mathcal{P}^*$.

THEOREM 2.3. *Suppose that we run the system with a (λ, w, p, M) adversary, where $\lambda < 1$ and $p > 2$, using protocol $\mathcal{P}^*$ (described in section 3). Let W_t be the wait of a job that arrived at time t. Then*

$$\limsup_{t\to\infty} \mathbf{E}[W_t \mid L_0] = O(\gamma^{-1}(w + \ln n)(1+M)^{3p}) \qquad \text{as } n \to \infty.$$

In addition, if the adversary is strongly bounded, then for any constant $c > 0$ there is a constant $\kappa = \kappa(c)$, such that for sufficiently large n,

$$\liminf_{t\to\infty} \mathbf{Pr}(W_t \le \kappa\gamma^{-1}(w + \ln n)) \ge 1 - n^{-c}.$$

The above bounds hold without the limits if the system starts with no load.

In particular, for bounded degree regular expanders, $\gamma = \Theta(1)$, and the diameter of the graph is $\Theta(\ln n)$, so for these graphs the above result is optimal.

As a special case of a strongly bounded adversary, we consider the generator model that appeared in [4]. We contrast our results with the work stealing load balancing protocol that is analyzed there and show that in sparse networks, both the load in the system and the waiting time of a job can be exponential in the network size.

Notice that both Theorems 2.2 and 2.3 are concerned with the long-term behavior of the system. In both cases, we show that if certain restrictions are placed on the adversary, then the system behaves well most of the time. Since we focus on the asymptotic behavior of the system, every valid system state will occur infinitely often. Therefore the analysis must ensure that the system rarely enters undesirable states and quickly recovers from them when it does. Simple probabilistic techniques (e.g., estimating the probability of rare events for every time point and using a union bound over all time steps) are not sufficient to achieve these goals. We require more sophisticated arguments.

There is a substantial class of results, based on modeling the system's evolution as a Markov chain, that is frequently employed in these sorts of analyses—the proof of stability for the work stealing protocol in [4] is one example. Furthermore, for this Markov chain setting, there are additional tools for proving rapid convergence to a stationary distribution (see Meyn and Tweedie [12]), as well as for deriving properties of the stationary distribution (see [4] and the references therein). However, many of these results are qualitative and not quantitative. More importantly, they restrict the class of problems that they can model. In the particular context of adversarial load balancing that is the focus of this work, any straightforward application of such tools will only yield results for Markovian adversaries (i.e., adversaries whose usage allows for the system to be modeled as a time-homogeneous Markov chain)—the analysis in [4] is an example. While it is not obvious whether this restricted class of adversaries is really any weaker than the general class presented earlier, it is clear that results in the general adversarial model serve as a compelling argument for the efficacy of the load balancing protocol under a wide range of input conditions.

Since we seek greater generality than Markov chain results are known to provide, we need more elaborate tools. In the conference version of this paper [2], we apply results from renewal theory to show that the system has efficient long-term behavior under a particular non-Markovian adversary. In the current work, we generalize the results presented there by analyzing the behavior under the more general adversaries of Definition 2.1. To this end, we show that the load in the system behaves like a supermartingale above some threshold and then employ a result of Pemantle and Rosenthal [17] for the analysis of such processes. Finally, in section 5.3, we derive stronger, high-probability results for the more restricted adversaries.

3. Protocol. If we were simply interested in a stability result we could use the protocol studied in [8]: in each step nodes are matched randomly with adjacent neighbors in the network, and if node v is matched with node u, they equalize their load subject to integer rounding. The details of the protocol (which we call protocol $\mathcal{P}$) are given below.

```
1. Matching phase:
  • For each node $i$
    – Node $i$ inserts each incident edge $(i,j)$ into a set $S$ with probability $\frac{1}{8\max(d_i,d_j)}$, where
      $d_i$ is the degree of node $i$.
    – Node $i$ removes edge $(i,j)$ from $S$ if some edge $(i,k)$ or $(j,k)$ is in $S$, with $k \neq i,j$.
  • Let the matching $M$ consist of the remaining edges in $S$.
2. Transfer phase:
  • If $(i,j) \in M$
    – $i$ and $j$ equalize their loads so that, say, $i$ gets load $\lceil(\ell_t(i)+\ell_t(j))/2\rceil$ and $j$ gets load
      $\lfloor(\ell_t(i)+\ell_t(j))/2\rfloor$, where $\ell_t(i)$ is the load of processor $i$ in the beginning of the step.
```

To bound the waiting time of jobs in the system we need to augment the above protocol with a distributed version of the *first-in-first-out* queueing discipline. It is not enough to require that a node consume the oldest job in its queue; we also need to consider the jobs' ages in the load balancing procedure. In particular, when u and v are matched they should not only equalize their total load but also equalize (up to rounding) the load that they have above any given age. A simple method to maintain this property is given by protocol $\mathcal{P}^*$ below. Note that protocol $\mathcal{P}^*$ is a special case of protocol $\mathcal{P}$.

```
1. Matching phase:
  • For each node $i$
    – Node $i$ inserts each incident edge $(i,j)$ into a set $S$ with probability $\frac{1}{8\max(d_i,d_j)}$, where
      $d_i$ is the degree of node $i$.
    – Node $i$ removes edge $(i,j)$ from $S$ if some edge $(i,k)$ or $(j,k)$ is in $S$, with $k \neq i,j$.
  • Let the matching $M$ consist of the remaining edges in $S$.
2. Transfer phase:
  • If $(i,j) \in M$
    – Let $J_1^i, J_2^i, \ldots, J_{\ell_t(i)}^i$ (where $\ell_t(i)$ is the load of processor $i$ in the beginning of the step),
      and let $J_1^j, J_2^j, \ldots, J_{\ell_t(j)}^j$ be the jobs in the queues of nodes $i$ and $j$, respectively, sorted
      from oldest to newest.
    – Node $i$ sends jobs $J_2^i, J_4^i, \ldots, J_{2\lfloor \ell_t(i)/2\rfloor}^i$ to node $j$. Similarly, node $j$ sends jobs
      $J_2^j, J_4^j, \ldots, J_{2\lfloor \ell_t(i)/2\rfloor}^j$ to node $i$.
    – Node $i$ merges jobs $J_1^i, J_3^i, J_5^i, \ldots$ with $J_2^j, J_4^j, J_6^j, \ldots$ in its queue, so that, finally, if a
      job $J$ is older than a job $J'$, then job $J$ is in the queue before job $J'$.
    – Similarly, node $j$ merges jobs $J_1^j, J_3^j, J_5^j, \ldots$ with $J_2^i, J_4^i, J_6^i, \ldots$ in its queue, so that,
      finally, if a job $J$ is older than a job $J'$, then job $J$ is in the queue before job $J'$.
```

4. Analysis of the static case. We first analyze our load balancing protocol in a static setting in which some initial load is placed on the n processors, and load is moved between processors until the loads in all the processors are approximately equal. No new load is added to or removed from the system throughout the execution of the protocol.

Our analysis of the static case is based on coupling the execution of our protocol with the nonintegral protocol studied in [8]. The only difference between the two protocols is that in the nonintegral protocol the load is equally distributed between the two matched processors with no rounding. We consider two copies of the network starting with the same initial distribution, one using our protocol and the other using the nonintegral protocol. The two processes are coupled so that they use the same matching at every time step.

Fix any initial distribution of K tokens to the nodes in V, and let $\bar{\ell} = K/n$. For each time step $t \geq 0$ and $u \in V$, let $\ell_t(u)$ be the number of tokens at u at the end of step t of the original, integral protocol, and let $\ell'_t(u)$ be the number of tokens at u at the end of step t in the nonintegral copy of the protocol. Also, for each time step $t \geq 0$, let $\Phi'_t = \sum_{v \in V}(\ell'_t(v) - \bar{\ell})^2$. Note that if the total load in the system is K, then for any $t \geq 0$, $\Phi'_t \leq K^2$. Notice that Φ'_t corresponds to the variance of the load on the nodes, and it is easy to see that it is nonincreasing with time, which, intuitively, means that successive applications of the load balancing protocol even out the distribution of the tokens to the nodes.

Recall that $\gamma = \Lambda/16\Delta$. The performance of the nonintegral protocol was analyzed in terms of γ in [8]. The relevant result is the following lemma, which follows immediately from Theorem 1 in that work.

LEMMA 4.1. *For any $t \geq 0$,*

$$\mathbf{E}[\Phi'_{t+1} \,|\, \Phi'_t] \leq (1-\gamma)\Phi'_t.$$

Adapting the technique in [8] we can prove the following static load balancing result for the nonintegral copy.

LEMMA 4.2. *If $t \geq \gamma^{-1}(2\ln K + c\ln n)$, then the probability that there is some $v \in V$ with $|\ell'_t(v) - \bar{\ell}| > 1$ is at most $1/n^c$.*

Proof. By Lemma 4.1 we get

$$\mathbf{E}[\Phi'_t] = \mathbf{E}[\,\mathbf{E}[\Phi'_t \,|\, \Phi'_{t-1}]\,] \leq (1-\gamma)\mathbf{E}[\Phi'_{t-1}].$$

By applying the same argument t times we get

$$\begin{aligned}\mathbf{E}[\Phi'_t] &= (1-\gamma)^t\Phi'_0 \\ &\leq \Phi'_0 e^{-\gamma t} \\ &\leq e^{-c\ln n},\end{aligned}$$

where in the last inequality we used the facts that $\Phi'_0 \leq K^2$ and $t \geq \gamma^{-1}(2\ln K + c\ln n)$. Applying Markov's inequality yields $\mathbf{Pr}(\Phi'_t > 1) < n^{-c}$. □

We will now tie the performance of our protocol to the performance of the nonintegral copy.

LEMMA 4.3. *For any $t \geq 0$ and any $v \in V$, $|\ell_t(v) - \ell'_t(v)| \leq t/2$, regardless of the chosen matchings and the rounding decisions in the original protocol.*

Proof. The proof is by induction on $t \geq 0$. If $t = 0$, then $\ell_t(v) = \ell'_t(v)$ for every $v \in V$, because no randomness has been introduced into the process yet. For the induction step, suppose that the lemma holds for time t. Choose any $v \in V$. If v is not matched at time $t+1$, then $\ell_{t+1}(v) = \ell_t(v)$ and $\ell'_{t+1}(v) = \ell'_t(v)$, so the lemma holds by the induction hypothesis. Otherwise, v is matched to some vertex u at time $t+1$. In this case, for any rounding choice, $(\ell_t(u)+\ell_t(v)-1)/2 \le \ell_{t+1}(v) \le (\ell_t(u)+\ell_t(v)+1)/2$. Also, $\ell'_{t+1}(v) = (\ell'_t(u) + \ell'_t(v))/2$. These observations give

$$\begin{aligned}|\ell_{t+1}(v) - \ell'_{t+1}(v)| &\overset{(a)}{\le} \frac{1}{2} + \left|\frac{\ell_t(u)+\ell_t(v)}{2} - \frac{\ell'_t(u)+\ell'_t(v)}{2}\right| \\ &\overset{(b)}{\le} \frac{1}{2} + \frac{1}{2}|\ell_t(u) - \ell'_t(u)| + \frac{1}{2}|\ell_t(v) - \ell'_t(v)| \\ &\overset{(c)}{\le} \frac{1}{2}\cdot\frac{t}{2} + \frac{1}{2}\cdot\frac{t}{2} + \frac{1}{2} \\ &= \frac{t+1}{2},\end{aligned}$$

where (a) follows from previous observations, (b) follows from the triangle inequality, and (c) follows from the induction hypothesis. □

Putting Lemmas 4.2 and 4.3 together yields the following theorem.

THEOREM 4.4. *Let $\hat{t} = \gamma^{-1}(2\ln K + c\ln n)$. Then, for any $t \ge \hat{t}$, the probability that there is some $v \in V$ with $|\ell_t(v) - \bar{\ell}| > 1 + \frac{\hat{t}}{2}$ is at most n^{-c}.*

Proof. We first prove the theorem for the case $t = \hat{t}$. Regardless of the matchings generated in the first $\hat{t}$ steps, $|\ell_{\hat{t}}(v) - \ell'_{\hat{t}}(v)| \le \hat{t}/2$ for every $v \in V$ by Lemma 4.3. With probability at least $1 - 1/n^c$, $|\ell'_{\hat{t}}(v) - \bar{\ell}| \le 1$ for all $v \in V$ by Lemma 4.2. Adding these inequalities proves the theorem for the case $t = \hat{t}$. To extend the result for the case $t > \hat{t}$, notice that the sequence $\{\max_{v\in V}|\ell_t(v) - \bar{\ell}|\}_{t\ge\hat{t}}$ is nonincreasing. □

5. Analysis of the dynamic case. In order to analyze the long-term performance of the system, we split the time into *epochs* of a fixed length T_E (to be defined later). We analyze each epoch in Theorem 5.1, which is the key ingredient in showing the stability and waiting-time properties of the system. Notice that the first part of the theorem, which we will apply for the stability result, holds for any protocol obeying the rules of $\mathcal{P}$, while the second part, which will be applied for the waiting-time guarantees, uses protocol $\mathcal{P}^*$.

THEOREM 5.1. *For any constant $c > 0$ and load $\Theta = \Theta_n > 0$, consider an epoch of length $T_E = T_D + T_C$, such that*

$$T_D \ge \gamma^{-1}(2\ln\Theta + c\ln n) \qquad \textit{and} \qquad T_C \ge \frac{\Theta}{n} + \frac{T_D}{2} + 1.$$

1. *Running protocol $\mathcal{P}$, if at time τ the load is L_τ, then the system consumes at least $\min\{L_\tau, \Theta\}$ tokens in the next T_E steps with probability at least $1-n^{-c}$.*
2. *Running protocol $\mathcal{P}^*$, if at time τ the load is bounded by Θ, then with probability at least $1 - n^{-c}$, all the jobs that exist in the system at time τ will be consumed by time $\tau + T_E$.*

Proof. For the purpose of the analysis we split the epoch into two parts. In the first T_D steps we focus on the distribution of load between the processors, and in the remaining T_C steps we focus on the consumption of load by the processors (although load is consumed throughout the whole execution by processors that have load).

We start by proving the first part of the theorem; a modification of that argument gives the second part. To analyze the distribution of load between the processors, we couple the actual execution of the protocol in the first T_D steps with an execution of the protocol in a static setting that starts with a total load of exactly Θ and does not generate or consume any jobs. We refer to the actual execution of the protocol as the *dynamic copy* and the static execution as the *static copy*.

To formulate the coupling we color all the tokens (jobs) in the dynamic copy at time τ by red and blue. A subset of $M = \min\{L_\tau, \Theta\}$ tokens in the system at time τ is colored red, and the rest are colored blue. We now place Θ tokens in the static copy so that each node in the static copy starts the process with at least as many tokens as the number of red tokens in the corresponding node of the dynamic copy. New tokens that arrive through the execution of the dynamic copy are colored blue.

The executions of the two copies are coupled so that they use the same matching in each step and the same rounding decisions. When we equalize (up to one) the load between two vertices in the dynamic execution, we also equalize (up to one, using the same rounding rule) the number of red tokens in the two nodes, keeping the total number of red tokens in the two nodes as before (this can be achieved by recoloring some tokens). Finally, after each matching we recolor the tokens in the nodes again, preserving the total number of red tokens in each queue, but putting all the red tokens in a queue ahead of all the blue tokens.

LEMMA 5.2. *At any time $\tau \le t \le \tau + T_D$ the number of red tokens in each node of the dynamic copy is bounded by the number of tokens in the corresponding node in the static copy.*

Proof. We begin with some intuition. Initially the coloring of the tokens in the dynamic copy is such that the claim holds. New tokens that enter the system in the dynamic copy are colored blue, while no tokens in the static copy are removed. Furthermore, the matching operations in both copies are coupled, so we expect that if one node in the static copy gives half of its tokens to a neighbor, the same node in the dynamic copy will do the same with its red tokens. Therefore, we expect that the claim will hold at the next step. By induction, the lemma follows.

We now proceed formally by induction on t. The claim is true for $t = \tau$ by the construction of the static copy. Assume that the claim holds after the execution of step $t - 1$, and consider the load of node u after the execution of step t. If u was not part of a matching in step t, then the load of the static copy did not change. The number of red tokens of the dynamic copy either did not change or was reduced by 1 if a red token was consumed. In both cases, using the induction hypothesis, the number of red tokens in the dynamic copy after the execution of step t is bounded by the number of tokens in the static copy of u.

Assume now that node u was matched with node v during step t. Recall that every time step in the process can be decomposed into three substeps: the insertion substep (where new load is created in the dynamic copy), the balancing substep (where the matching is chosen and the nodes in both copies equalize their loads), and the consumption substep (where the nonempty nodes in the dynamic copy consume a job). With this in mind, we define some new variables:

- $\ell_t^i(u)$, $\ell_t^b(u)$, $\ell_t^c(u)$ are the total number of tokens at node u in the dynamic copy immediately following the insertion, balancing, and consumption substeps of time step t, respectively.
- $r_t^i(u)$, $r_t^b(u)$, $r_t^c(u)$ are the number of red tokens at node u in the dynamic copy at that time.
- $s_t^i(u)$, $s_t^b(u)$, $s_t^c(u)$ are the number of tokens at node u in the static copy at that time.

We analyze each substep separately.

• Initially, by the induction hypothesis, we have

$$r^c_{t-1}(u) \le s^c_{t-1}(u), \qquad r^c_{t-1}(v) \le s^c_{t-1}(v). \tag{5.1}$$

• After the insertion substep, we color the new tokens blue so the number of red tokens remains the same:

$$r^i_t(u) = r^c_{t-1}(u), \qquad r^i_t(v) = r^c_{t-1}(v). \tag{5.2}$$

The static copy does not accept new tokens, so

$$s^i_t(u) = s^c_{t-1}(u), \qquad s^i_t(v) = s^c_{t-1}(v). \tag{5.3}$$

So from relations (5.1), (5.2), and (5.3) we get that

$$r^i_t(u) \le s^i_t(u), \qquad r^i_t(v) \le s^i_t(v). \tag{5.4}$$

Notice also that the number of red tokens is bounded by the total number of tokens, so

$$r^i_t(u) \le \ell^i_t(u), \qquad r^i_t(v) \le \ell^i_t(v). \tag{5.5}$$

• After the balancing substep, we assume, without loss of generality, that the rounding is such that

$$\ell^b_t(u) = \left\lceil \frac{1}{2}(\ell^i_t(u) + \ell^i_t(v)) \right\rceil, \qquad \ell^b_t(v) = \left\lfloor \frac{1}{2}(\ell^i_t(u) + \ell^i_t(v)) \right\rfloor. \tag{5.6}$$

The results of the analysis of the static case in section 4 hold for an arbitrary rounding procedure. Thus, in the static copy we perform the rounding so that

$$s^b_t(u) = \left\lceil \frac{1}{2}(s^i_t(u) + s^i_t(v)) \right\rceil, \qquad s^b_t(v) = \left\lfloor \frac{1}{2}(s^i_t(u) + s^i_t(v)) \right\rfloor. \tag{5.7}$$

Finally, we recolor the tokens in the dynamic copy (we swap colors between some tokens), so that

$$r^b_t(u) = \left\lceil \frac{1}{2}(r^i_t(u) + r^i_t(v)) \right\rceil, \qquad r^b_t(v) = \left\lfloor \frac{1}{2}(r^i_t(u) + r^i_t(v)) \right\rfloor. \tag{5.8}$$

Then, using relation (5.4), we get

$$r^b_t(u) = \left\lceil \frac{1}{2}(r^i_t(u) + r^i_t(v)) \right\rceil \le \left\lceil \frac{1}{2}(s^i_t(u) + s^i_t(v)) \right\rceil = s^b_t(u) \tag{5.9}$$

and

$$r^b_t(v) = \left\lfloor \frac{1}{2}(r^i_t(u) + r^i_t(v)) \right\rfloor \le \left\lfloor \frac{1}{2}(s^i_t(u) + s^i_t(v)) \right\rfloor = s^b_t(v). \tag{5.10}$$

Notice that

$$r^b_t(u) + r^b_t(v) = \left\lceil \frac{1}{2}(r^i_t(u) + r^i_t(v)) \right\rceil + \left\lfloor \frac{1}{2}(r^i_t(u) + r^i_t(v)) \right\rfloor = r^i_t(u) + r^i_t(v),$$

which means that we haven't changed the number of red tokens, and notice also that by using relation (5.5) we get that

$$r_t^b(u) = \left\lceil \frac{1}{2}(r_t^i(u) + r_t^i(v)) \right\rceil \leq \left\lceil \frac{1}{2}(\ell_t^i(u) + \ell_t^i(v)) \right\rceil = \ell_t^b(u)$$

and

$$r_t^b(v) = \left\lfloor \frac{1}{2}(r_t^i(u) + r_t^i(v)) \right\rfloor \leq \left\lfloor \frac{1}{2}(\ell_t^i(u) + \ell_t^i(v)) \right\rfloor = \ell_t^b(v),$$

which ensure that the recoloring is valid (i.e., for both u and v, the number of the red tokens is not more than the total number of tokens).

Finally, we recolor the tokens again (preserving the number of red tokens at each node), so that every red token in a queue is ahead of every blue token in that queue.

• After the consumption substep, since some red tokens may be consumed, we get that

$$r_t^c(u) \leq r_t^b(u), \qquad r_t^c(v) \leq r_t^b(v). \tag{5.11}$$

In the static copy there are no tokens being consumed, so

$$s_t^c(u) = s_t^b(u), \qquad s_t^c(v) = s_t^b(v). \tag{5.12}$$

Hence, from relations (5.9), (5.10), (5.11), and (5.12) we can finally prove the induction hypothesis for step t:

$$r_t^c(u) \leq s_t^c(u), \qquad r_t^c(v) \leq s_t^c(v). \qquad \square$$

We now turn to studying the consumption phase. Applying Theorem 4.4 to the execution of the static copy, we see that at time $\tau + T_D$ (the reader can verify that the condition of Theorem 4.4 holds for $t = T_D$ and $K = \Theta$), with probability at least $1 - n^{-c}$ no node of the static copy has more than $T_D/2 + \Theta/n + 1 \leq T_C$ tokens. Thus, by Lemma 5.2, with the same probability, no node in the dynamic copy has more than T_C red tokens.

We continue to run the coupling from time $\tau + T_D$ with this new coloring. When balancing between two nodes, we recolor the tokens in exactly the same way as in the distribution phase (see Lemma 5.2), so that, in particular, if a node has some red tokens in its queue, then these tokens are at the front of the node's queue, before the blue tokens. In this case, notice that after a balancing substep, the maximum (over all the nodes in the graph) number of tokens does not increase. In the consumption substep the maximum (again, over all the nodes in the graph) number of red tokens decreases by 1, since the red tokens are at the front of their queues. Since initially at most T_C red tokens are at a node with probability at least $1 - n^{-c}$, we conclude that with probability $1 - n^{-c}$, all the red tokens (which means at least $M = \min\{L_\tau, \Theta\}$ tokens) are consumed in this epoch. This completes the proof of the first part of the theorem.

The proof of the second part of the theorem is almost identical to the first one. We color initially all the $L_\tau \leq \Theta$ tokens of the dynamic copy at time τ red, while all the subsequent ones are colored blue, and again we consider the coupled static copy. Since now we are interested in the identities of the jobs that are being consumed, we do not allow recolorings of the tokens.

Notice though that protocol $\mathcal{P}^*$ ensures that during the whole epoch, if a node has both red and blue tokens in its queue, the red tokens are before the blue ones. Moreover, the transfer phase of $\mathcal{P}^*$ ensures that when node u is matched with node v, they balance their red tokens (up to a difference of 1). Without loss of generality, we assume that the rounding is such that if the ensemble of the red tokens in the two nodes is odd, then node u ends up with one more red token. Hence equations (5.8) remain true. We then perform the rounding in the static copy to ensure that equations (5.7) remain true as well.

As an aside, notice that it may be the case that after the balancing substep, node v's *total* load is exactly one larger than u's total load, in which case equations (5.6) may become

$$\ell_t^b(u) = \left\lfloor \frac{1}{2}(\ell_t^i(u) + \ell_t^i(v)) \right\rfloor, \qquad \ell_t^b(v) = \left\lceil \frac{1}{2}(\ell_t^i(u) + \ell_t^i(v)) \right\rceil.$$

Here, however, we analyze only the distribution of the red tokens, so the analysis is not affected by this fact.

Everything else is identical to the first part, and by the same reasoning we conclude that by the end of the distribution phase, for every node u, the number of red tokens in the dynamic copy is bounded by the number of tokens in the static copy, with probability at least $1 - n^{-c}$, which, by applying Theorem 4.4, implies that every node has at most T_C red tokens.

Then, as in the first part, we can conclude that by the end of the consumption phase all the red tokens—which are exactly the tokens that were in the system in the beginning of the epoch—are consumed. □

An immediate consequence of the analysis of the second part of Theorem 5.1 is the following lemma, which gives a bound on the expected time needed until the initial load is distributed. We make use of the lemma in order to bound the expected waiting time.

LEMMA 5.3. *Assume that we are given c, Θ, T_D, T_C, satisfying the conditions of Theorem* 5.1*, and assume that we balance according to protocol $\mathcal{P}^*$. If at time τ the load is bounded by Θ, then the expected time needed until every node in the system has at most T_C of the initial jobs is bounded by $2T_D$ for sufficiently large n. At every time, when a node has some of the initial load, this is at the head of its queue.*

Proof. The analysis is an extension of the proof of the second part of Theorem 5.1. We color the initial load at time τ red, all the new incoming load blue, and we let T be the time until every node in the network has at most T_C red tokens. We perform the same coupling as in Theorem 5.1 from time τ until time $\tau + T$. During this whole period the number of red tokens in the dynamic copy is bounded by the number of tokens in the static copy. Since the red tokens are the oldest tokens in the system, protocol $\mathcal{P}^*$ ensures that if a node has some red tokens and some blue tokens, the red tokens are always in front of the blue ones.

Thus, by the proof of Theorem 5.1, part 2, we get that $T \le T_C$ with probability at least $1 - n^{-c}$. It follows that $\lceil T/T_D \rceil$ is stochastically dominated by a geometric random variable with parameter $1 - n^{-c}$, so

$$\mathbf{E}[T] \le \frac{1}{1 - n^{-c}} T_D \le 2T_D$$

for sufficiently large n. □

5.1. Stability. In this section we prove the stability of the system under a $(\lambda < 1, w, p > 2, M)$ stochastic adversary. The main technical tool that we use is the following theorem, which follows immediately from [17, Corollary 2].

THEOREM 5.4. *Let $X_1, X_2, \ldots$ be a sequence of nonnegative random variables satisfying the following conditions:*

1. *There exist positive numbers $\alpha = \alpha_n$ and $\Theta = \Theta_n$ such that for all $x_1, \ldots, x_i$ with $x_i > \Theta$,*
$$\mathbf{E}[X_{i+1} - X_i \,|\, X_1 = x_1, \ldots, X_i = x_i] \leq -\alpha.$$
2. *There exists a positive number $\xi = \xi_n$ and a $p = p_n > 2$ such that for all $x_1, \ldots, x_i$*
$$\mathbf{E}[|X_{i+1} - X_i|^p \,|\, X_1 = x_1, \ldots, X_i = x_i] \leq \xi.$$

Then there exists $\Xi = \Xi(X_0, \alpha, \Theta, \xi)$ and t_0 such that for all $t \geq t_0$,
$$\mathbf{E}[X_t \,|\, X_0] \leq \Xi + \max(0, X_0 - \Theta).$$
Furthermore, assuming that p is a constant with respect to n,
$$\Xi = O\left(\Theta + \alpha\left(1 + \frac{\xi}{\alpha^p}\right)^{3p}\right) \qquad \textit{as } n \to \infty.$$

Our stability result is summarized in the following theorem.

THEOREM 5.5. *Suppose that we run protocol $\mathcal{P}$ with a (λ, w, p, M) adversary, where $\lambda < 1$ and $p > 2$. Then*
$$\sup_{t \geq 0} \mathbf{E}[L_t \,|\, L_0] = O(\max(\gamma^{-1} n(w + \ln n)(1 + M)^{3p}, L_0)) \qquad \textit{as } n \to \infty.$$

Proof. We first present the high-level idea of the proof. Recall that L_t is the load of the system at time t. We partition the time into *epochs* of some length d and apply Theorem 5.4 to the subsequence $\{L_{di} \mid i \geq 0\}$. Using Theorem 5.1, we show that if the load at the beginning of an epoch is above some threshold Θ, then the expected load at the end of that epoch is strictly smaller by a significant amount, proving condition 1 of Theorem 5.4. We then derive the required bound on the pth moment $L_{(i+1)d} - L_{id}$, establishing condition 2 of Theorem 5.4. Applying Theorem 5.4 then gives the desired bound on the maximum expected load at the end of epochs. Finally, we generalize to steps that are not multiples of d.

Let us now formalize the above argument. The two conditions that we want to satisfy are
$$\mathbf{E}[L_{(i+1)d} - L_{id} \,|\, L_0 = l_0, L_d = l_d, \ldots, L_{(i-1)d} = l_{(i-1)d}, L_{id} = \ell_{id} > \Theta] \leq -\alpha,$$
$$\mathbf{E}[|L_{(i+1)d} - L_{id}|^p \,|\, L_0 = l_0, L_d = l_d, \ldots, L_{(i-1)d} = l_{(i-1)d}, L_{id} = \ell_{id}] \leq \xi$$
for all $(l_0, l_1, \ldots, l_{id})$. In order to simplify the notation we denote
$$\mathcal{L}_{i-1} = \{L_0 = l_0, L_d = l_d, \ldots, L_{(i-1)d} = l_{(i-1)d}\},$$
so the conditions become
$$\mathbf{E}[L_{(i+1)d} - L_{id} \,|\, \mathcal{L}_{i-1}, L_{id} = \ell_{id} > \Theta] \leq -\alpha, \tag{5.13}$$
$$\mathbf{E}[|L_{(i+1)d} - L_{id}|^p \,|\, \mathcal{L}_{i-1}, L_{id}] \leq \xi. \tag{5.14}$$

Let

$$\Theta = \sigma\gamma^{-1}n(w + \ln n) \tag{5.15}$$

for some constant σ that we will fix later. Also let

$$T_D = \gamma^{-1}(2\ln\Theta + \ln n) \qquad \text{and} \qquad T_C = \frac{\Theta}{n} + \frac{T_D}{2} + 1,$$

so that

$$\begin{aligned} T_E &= T_D + T_C \\ &= \frac{\Theta}{n} + 3\gamma^{-1}\ln\Theta + \frac{3}{2}\gamma^{-1}\ln n + 1 \\ &= \sigma\gamma^{-1}(w + \ln n) + 3\gamma^{-1}\ln\sigma + 3\gamma^{-1}\ln\gamma^{-1} + 3\gamma^{-1}\ln(w + \ln n) + \frac{9}{2}\gamma^{-1}\ln n + 1. \end{aligned} \tag{5.16}$$

We let σ be such that T_E/w is an integer, say k (we show later that such a σ exists), so that T_E is a multiple of the window size w, and define the epoch length

$$d = kw = T_E. \tag{5.17}$$

Notice that $d = O(\gamma^{-1}(w + \ln n))$, a fact that we use later. Fix some time $t = id$ and consider what happens in the case that $L_t > \Theta$. By Theorem 5.1, part 1 (for $c = 1$), we get that within $d = T_E$ steps the system consumes at least Θ units of load, with probability at least $1 - 1/n$. Thus, the expected number of tokens consumed between steps id and $(i+1)d$ is at least

$$\left(1 - \frac{1}{n}\right) \cdot \Theta = \sigma\gamma^{-1}n(w + \ln n) - \sigma\gamma^{-1}(w + \ln n), \tag{5.18}$$

independently of the past, and of any of the adversary's decisions.

Since an epoch consists of k windows, by the definition of the adversary, the expected injected new load in the system from time id until $(i+1)d$, conditioned on $\mathcal{L}_{i-1}$, is bounded by

$$k\lambda nw = \lambda nd.$$

Using (5.16) and (5.17) we can conclude that the expected new load injected by the adversary, conditioned on the history, is bounded by

$$\begin{aligned} \lambda nd &= \lambda\sigma\gamma^{-1}n(w + \ln n) + 3\lambda\gamma^{-1}n\ln\sigma \\ &\quad + 3\lambda\gamma^{-1}n\ln\gamma^{-1} + 3\lambda\gamma^{-1}n\ln(w + \ln n) + \frac{9}{2}\lambda\gamma^{-1}n\ln n + \lambda n \\ &\le \lambda\sigma\gamma^{-1}n(w + \ln n) + 3\lambda\gamma^{-1}n\ln\sigma \\ &\quad + 9\lambda\gamma^{-1}n\ln(c'n) + 3\lambda\gamma^{-1}n\ln(w + \ln n) + \frac{9}{2}\lambda\gamma^{-1}n\ln n + \lambda n \\ &= \gamma^{-1}n(w + \ln n)\left[\lambda\sigma + \frac{3\lambda\ln\sigma}{w + \ln n} + \frac{9\lambda\ln(c'n)}{w + \ln n} + \frac{3\lambda\ln(w + \ln n)}{w + \ln n} \right. \\ &\qquad \left. + \frac{9\lambda\ln n}{2(w + \ln n)} + \frac{\lambda}{\gamma^{-1}(w + \ln n)}\right], \end{aligned} \tag{5.19}$$

where the inequality follows from the fact that $\gamma^{-1} \leq c'n^3$ for some constant c', since $\Lambda = \Omega(n^{-2})$ for any connected graph. Therefore, from relations (5.18) and (5.19) we get

$$\begin{aligned}\mathbf{E}[L_{(i+1)d} - L_{id} \,|\, \mathcal{L}_{i-1}] &\leq -\sigma\gamma^{-1}n(w+\ln n) + \gamma^{-1}n(w+\ln n) \\ &\quad \left[\lambda\sigma + \frac{3\lambda\ln\sigma}{w+\ln n} + \frac{9\lambda\ln(c'n)}{w+\ln n} + \frac{3\lambda\ln(w+\ln n)}{w+\ln n}\right. \\ &\quad \left. + \frac{9\lambda\ln n}{2(w+\ln n)} + \frac{\lambda}{\gamma^{-1}(w+\ln n)} + \frac{\sigma}{n}\right] \\ &\leq -\sigma\gamma^{-1}n(w+\ln n) + \gamma^{-1}n(w+\ln n)(\lambda\sigma+Q),\end{aligned}$$

for sufficiently large n, where Q is a constant independent of σ. Hence,

$$\mathbf{E}[L_{(i+1)d} - L_{id} \,|\, \mathcal{L}_{i-1}] \leq -\gamma^{-1}n(w+\ln n)(\sigma - \lambda\sigma - Q).$$

If

$$\sigma > \frac{Q}{1-\lambda},$$

then relation (5.13) is satisfied for sufficiently large n. We therefore let σ be the smallest number that is greater than $Q/(1-\lambda)$ that ensures that $d = T_E$ is a multiple of the window size w. Notice that the fact that d is a continuous function of σ ensures that such a value of σ exists and satisfies

$$\sigma \leq \frac{2Q}{1-\lambda}.$$

Thus condition 1 is satisfied for $\alpha = O(\gamma^{-1}n(w+\ln n))$.

We now turn our attention to relation (5.14). Let J_i and Z_i be the number of tokens injected by the adversary and consumed by the processors, respectively, during the ith epoch. We note that

$$\begin{aligned}|L_{(i+1)d} - L_{id}|^p &= |J_i - Z_i|^p \\ &\leq J_i^p + Z_i^p \\ &\leq J_i^p + d^p n^p,\end{aligned} \tag{5.20}$$

where the last inequality follows from the fact that the system (deterministically) consumes at most n tokens in every step.

We now bound $\mathbf{E}[J_i^p \,|\, \mathcal{L}_{i-1}, L_{id}]$. Write $J_i = \sum_{j=0}^{k-1} Y_j$, where

$$Y_j = \sum_{t=0}^{w-1} I_{id+jw+t}$$

is the number of tokens injected by the adversary during the window $[id + jw, id + (j+1)w - 1]$.

The second condition on the stochastic adversary gives, for all j,

$$\mathbf{E}[Y_j^p] \leq Mn^p w^p.$$

Applying Hölder's inequality gives

$$J_i^p = \left(\sum_{j=0}^{k-1} Y_j\right)^p \leq \left(\sum_{j=0}^{k-1} 1\right)^{p\left(1-\frac{1}{p}\right)} \left(\sum_{j=0}^{k-1} Y_j^p\right) = k^{p-1}\sum_{j=0}^{k-1} Y_j^p,$$

and therefore we get

$$\mathbf{E}[J_i^p \,|\, \mathcal{L}_{i-1}, L_{id}] \leq k^p M n^p w^p. \tag{5.21}$$

Therefore, relations (5.20) and (5.21) imply that relation (5.14) is satisfied with

$$\xi = k^p M n^p w^p + d^p n^p = (M+1) d^p n^p,$$

and by Theorem 5.4 we deduce that

$$\sup_{i \geq 0} \mathbf{E}[L_{id} \,|\, L_0] = O(\max(\gamma^{-1} n(w + \ln n)(1+M)^{3p}, L_0)).$$

We have now proven the theorem for t corresponding to the beginning of an epoch. To finish the proof, we have for any $t \geq 0$

$$\begin{aligned}\mathbf{E}[L_t \,|\, L_0] &= \mathbf{E}[L_{\lfloor t/d \rfloor d + (t - \lfloor t/d \rfloor d)} \,|\, L_0] \\ &\leq \mathbf{E}[L_{\lfloor t/d \rfloor d} \,|\, L_0] + \mathbf{E}\left[\sum_{j = \lfloor t/d \rfloor d}^{t} I_j \,\middle|\, L_0 \right] \\ &\leq \sup_{i \geq 0} \mathbf{E}[L_{id} \,|\, L_0] + \lambda n d \\ &= O(\max(\gamma^{-1} n(w + \ln n)(1+M)^{3p}, L_0)). \qquad \square\end{aligned}$$

THEOREM 5.6. *Given any initial load L_0, with probability 1 there is an $i \geq 0$, such that $L_{id} \leq \Theta$ for Θ defined as in (5.15).*

Proof. For intuition, assume that the initial load L_0 is above Θ. Then by relation (5.13), we expect it to decrease below Θ after a sufficiently long time period. In order to prove this fact, we use martingale techniques. Since the expected load decreases independently of the past, we can couple the load with a supermartingale until it drops below Θ. Then we can apply a martingale convergence theorem to show that the supermartingale (and therefore the coupled system load) will eventually reach Θ.

Proceeding formally, we define a supermartingale $\{Y_{id} \mid i \geq 0\}$ with respect to the sequence $\{L_{id} \mid i \geq 0\}$, where

$$Y_{id} = \begin{cases} \max(L_0, \Theta) & \text{for } i = 0, \\ L_{id} & \text{if } L_{(i-1)d} > \Theta, \\ \Theta & \text{if } L_{(i-1)d} \leq \Theta. \end{cases}$$

As long as $Y_{(i-1)d} > \Theta$ the two sequences of random variables are identical. The sequence L_{id} assumes a value $\leq \Theta$ if and only if there is an index j such that $Y_{jd} = \Theta$.

The (nonnegative) supermartingale Y_{id} converges with probability 1 to a random variable Y (see [9, Theorem 5.1]), and since

$$\mathbf{E}[Y_{(i+1)d} \mid Y_{id} > \Theta] \leq Y_{id} - \alpha$$

for some $\alpha > 0$ (defined in the proof of Theorem 5.5), we have $\lim_{i \to \infty} \mathbf{E}[Y_{id}] = \Theta$. By applying Fatou's lemma [6, p. 110] we get $\mathbf{E}[Y] \leq \liminf_{i \to \infty} \mathbf{E}[Y_{id}] = \Theta$. Thus, with probability 1 the sequence $\{L_{id}\}$ assumes at some time jd a value less than or equal to Θ. $\square$

Corollary 5.7.

1. *If the system starts with no load, then for sufficiently large n,*

$$\sup_{t\geq 0} \mathbf{E}[L_t] = O(\gamma^{-1}n(w+\ln n)(1+M)^{3p}).$$

2. *For any starting conditions*

$$\limsup_{t\to\infty} \mathbf{E}[L_t \mid L_0] = O(\gamma^{-1}n(w+\ln n)(1+M)^{3p}) \qquad \text{as } n\to\infty.$$

Proof. The first part follows from Theorem 5.5, while the second part uses also Theorem 5.6. □

5.2. Waiting time. Having established that the system is stable, the next important performance parameter is the waiting time of a job from the time it enters the system until it is executed. For a given task that enters the system at time t, let W_t be the number of steps until the task is executed. Following the discussion of section 3, throughout this section we assume that we perform protocol $\mathcal{P}^*$.

Theorem 5.8. *Suppose that we run protocol $\mathcal{P}^*$ with a (λ, w, p, M) adversary, where $\lambda < 1$ and $p > 2$. Then*

$$\sup_{t\geq 0} \mathbf{E}[W_t \mid L_0] = O(\max(\gamma^{-1}(w+\ln n)(1+M)^{3p}, L_0/n)) \qquad \text{as } n\to\infty.$$

Proof. We begin with some intuition. By the results of section 5.1, we expect the load L_t at time t to be low, namely, bounded by $O(\max(\gamma^{-1}n(w+\ln n)(1+M)^{3p}, L_0))$. We also expect that the distribution protocol will rapidly distribute this load among the nodes (even if it is not already distributed, and regardless of any load that comes in after time t)—this is formalized by Lemma 5.3. Once this load is evenly distributed, it can be quickly consumed, since $\mathcal{P}^*$ ensures that every node consumes the oldest token in its queue at the end of every time step. Therefore, the expected time to consume all the load is $O(\mathbf{E}[L_t]/n)$.

We now proceed formally. Assume that at some time t the load in the system is L_t. We apply Lemma 5.3 with $c = 1$, $\Theta = L_t$, $T_D = \gamma^{-1}(2\ln L_t + \ln n)$ and

$$T_C = \frac{\Theta}{n} + \frac{T_D}{2} + 1,$$

and we get that the expected time, conditioned on any past event $\mathcal{H}$, until all the tokens that were present at time t are consumed (measured from time t) is at most

$$2T_D + T_C \leq 4T_C + T_C = 5T_C$$

for $n \geq 2$. Thus for $n \geq 2$,

$$\mathbf{E}[W_t \mid L_t, \mathcal{H}] \leq 5T_C = \frac{5L_t}{n} + \frac{5\gamma^{-1}}{2}(2\ln L_t + \ln n) + 5.$$

Consequently, for sufficiently large n, we have that for any $t \geq 0$,

$$\begin{aligned}\mathbf{E}[W_t \mid L_0] &= \sum_{\ell_t} \mathbf{E}[W_t \mid L_0, L_t = \ell_t] \cdot \mathbf{Pr}(L_t = \ell_t \mid L_0)\\ &\leq \sum_{\ell_t} \left(\frac{5\ell_t}{n} + \frac{5\gamma^{-1}}{2}(2\ln \ell_t + \ln n) + 5 \right) \cdot \mathbf{Pr}(L_t = \ell_t \mid L_0)\\ &= \mathbf{E}\left[\frac{5L_t}{n} + \frac{5\gamma^{-1}}{2}(2\ln L_t + \ln n) + 5 \,\middle|\, L_0 \right]\\ &= \frac{5\,\mathbf{E}[L_t \mid L_0]}{n} + \frac{5\gamma^{-1}}{2}(2\,\mathbf{E}[\ln L_t \mid L_0] + \ln n) + 5\\ &\leq \frac{5\,\mathbf{E}[L_t \mid L_0]}{n} + \frac{5\gamma^{-1}}{2}(2\,\ln \mathbf{E}[L_t \mid L_0] + \ln n) + 5\\ &= O(\max(\gamma^{-1}(w + \ln n)(1+M)^{3p}, L_0/n)),\end{aligned}$$

where the second-to-last step follows from Jensen's inequality applied to the concave function $f(x) = \ln x$, and the last step follows from Theorem 5.5. □

Applying Theorem 5.6 we have the following.

COROLLARY 5.9.

1. *If the system starts with no load, then for sufficiently large n,*

$$\sup_{t \geq 0} \mathbf{E}[W_t] = O(\gamma^{-1}(w + \ln n)(1+M)^{3p}).$$

2. *For any starting conditions,*

$$\limsup_{t \geq 0} \mathbf{E}[W_t \mid L_0] = O(\gamma^{-1}(w + \ln n)(1+M)^{3p}) \qquad \textit{as } n \to \infty.$$

Proof. The first part follows from Theorem 5.8, while the second part uses also Theorem 5.6. □

5.3. Strongly bounded adversaries. Recall that a strongly bounded adversary satisfies the additional requirement that for some constants $\alpha > 0$, $\beta \geq 1$, for any $\epsilon > 0$, the probability that the total number of new jobs that arrive in a given interval of length w is greater than $(1+\epsilon)\lambda n w$ is bounded by $e^{-\alpha \lambda n w \epsilon^{\beta}}$.

In this subsection we strengthen the preceding results for adversaries that are strongly bounded. The Chernoff-type restrictions on the input stream allow us to get high-probability results for the load and the waiting time.

5.3.1. High-probability bound on load. Our first result bounds the probability that at a given time point the load of the system is high.

THEOREM 5.10. *Consider a system where load is injected by a strongly bounded adversary. Let L_t be the load of the system at time t. Then for any sufficiently large constant c_1 there exists a constant $c' > 0$ such that*

$$\limsup_{t \to \infty} \mathbf{Pr}(L_t > c_1 \gamma^{-1} n (w + \ln n)) \leq 3n^{-c'}.$$

The above holds without the limit if the system starts with no load.

Proof. Let $\Theta = c_1 \gamma^{-1} n (w + \ln n)$. We observe the system at some time t, and we need to bound the probability that the load at time t is above Θ. Therefore, we assume that the load at time t is above Θ and calculate the probability of the events that may have led to such a load. If the load at some time $t' < t$ were smaller than Θ (which holds with probability 1 as $t \to \infty$ by Theorem 5.6), then from time t' up to t some rare events have taken place and increased the load much more than expected. We bound the probability of those events, thus bounding the probability that the load at time t is above Θ.

Similarly to Theorem 5.5, we split time into epochs of length

$$T_E = c_2\gamma^{-1}(w + \ln n)$$

starting from time t and going backwards. The constant c_2 (which depends on c_1) is chosen so that the epoch length is a multiple k of the window size ($T_E = kw$). Let $\mathcal{B}$ be the event $\{L_t \geq \Theta\}$, and for $i \leq t/T_E$ let $\mathcal{B}_i$ be the event that the load of the system is above Θ for exactly the last i epochs. More precisely,

$$\mathcal{B}_i = \{\forall j = 0, \ldots, i-1 :\ L_{t-jT_E} \geq \Theta, \quad L_{t-iT_E} < \Theta\}.$$

Let $\mathcal{C}_t$ be the event that the load in the system was not always above Θ before time t. Formally,

$$\mathcal{C}_t = \{\exists t' \leq t :\ L_{t'} \leq \Theta\}.$$

Then we have

$$\mathbf{Pr}(\mathcal{B}\,|\,\mathcal{C}_t) = \sum_{i=1}^{\lfloor t/T_E \rfloor} \mathbf{Pr}(\mathcal{B}_i \mid \mathcal{C}_t). \tag{5.22}$$

To estimate $\mathbf{Pr}(\mathcal{B}_i\,|\,\mathcal{C}_t)$ we distinguish between two cases, depending on the total load injected by the adversary during the i epochs immediately before t. Either the adversary inserted a lot of new jobs during this time, or he inserted a reasonable number of new jobs and the protocol failed to reduce the total load. Both cases are intuitively unlikely: the first by the strong bound on the adversary, and the second by the efficacy of the protocol. We bound the two cases separately and then use a union bound. We fix a constant $\epsilon > 0$ (whose actual value will be determined later) and define $\mathcal{M}$ as the event that "the injected load during the i epochs immediately preceding t is less than $K = (1+\epsilon)i\lambda n T_E = (1+\epsilon)i\lambda nkw$." Then we have

$$\mathbf{Pr}(\mathcal{B}_i\,|\,\mathcal{C}_t) \leq \mathbf{Pr}(\overline{\mathcal{M}}\,|\,\mathcal{C}_t) + \mathbf{Pr}(\mathcal{B}_i \cap \mathcal{M}\,|\,\mathcal{C}_t). \tag{5.23}$$

We bound each term separately, starting with $\mathbf{Pr}(\overline{\mathcal{M}}\,|\,\mathcal{C}_t)$. The first K jobs can be distributed in the ik windows of the period in

$$\binom{K + ik - 1}{ik} \tag{5.24}$$

ways.

We now bound the probability of each such distribution of the inserted jobs $(K_1, K_2, \ldots, K_{ik})$ (so that $\sum_{j=1}^{ik} K_j = K$). Recall that the expected number of jobs during the jth time window is at most λnw. Define ϵ_j as the deviation of K_j above λnw, namely,

$$\epsilon_j = \max\left(0, \frac{K}{\lambda nw} - 1\right).$$

In other words, ϵ_j is such that

$$K_j = (1+\epsilon_j)\lambda n w$$

if $K > \lambda n w$ and $\epsilon_j = 0$ otherwise. Since $\sum K_j = K$, we get that

$$\sum_{j=1}^{ik} \epsilon_j \geq ik\epsilon.$$

By using the definition of the strongly bounded adversary, we can bound the probability (conditioned on any past event) that in the jth window at least K_j were generated by

$$e^{-\alpha\lambda n w \epsilon_j^\beta}.$$

Therefore, the probability of a particular distribution $(K_1, \ldots, K_{id})$ of the first K jobs is bounded by

$$\prod e^{-\alpha\lambda n w \epsilon_j^\beta} = e^{-\alpha\lambda n w \sum \epsilon_j^\beta}.$$

Since $\beta \geq 1$, and $\sum \epsilon_j \geq ik\epsilon$, we obtain $\sum \epsilon_j^\beta \geq ik\epsilon^\beta$ (by raising to the βth power and using Hölder's inequality as in relation (5.21)), and the aforementioned probability becomes

$$e^{-\alpha\lambda n w i k \epsilon^\beta}.$$

Together with (5.24) we get that

$$\begin{aligned}
\mathbf{Pr}(\overline{\mathcal{M}} \,|\, \mathcal{C}_t) &\leq \binom{K+ik-1}{ik} e^{-\alpha\lambda n w i k\epsilon^\beta} \\
&< \binom{K+ik}{ik} e^{-\alpha\lambda n w i k\epsilon^\beta} \\
&\leq \left(\frac{(K+ik)e}{ik}\right)^{ik} e^{-\alpha\lambda n w i k\epsilon^\beta} \\
&= e^{ik\ln(K+ik)+ik-\alpha\lambda n w i k\epsilon^\beta - ik\ln(ik)} \\
&\leq e^{ik\ln[ik((1+\epsilon)\lambda n w+1)]+ik-\alpha\lambda n w i k\epsilon^\beta - ik\ln(ik)} \\
&= e^{ik\ln[(1+\epsilon)\lambda n w+1]+ik-\alpha\lambda n w i k\epsilon^\beta} \\
&< n^{-\kappa i}
\end{aligned} \tag{5.25}$$

for any constant κ and sufficiently large n.

Next we bound $\mathbf{Pr}(\mathcal{B}_i \cap \mathcal{M} \,|\, \mathcal{C}_t)$. By Theorem 5.1, part 1, if at the beginning of an epoch the load of the system is at least Θ, then the load decreases by at least Θ with probability at least $1 - n^{-c}$. This is the case for the last $i-1$ epochs. However, at time $t - iT_E$ the load of the system is below Θ, while at time t the load is above Θ. Moreover we have assumed that the total new load is at most $K = (1+\epsilon)i\lambda n T_E$. These facts imply that the consumed load is less than K, which in turn implies that in fewer than

$$\frac{K}{\Theta} = \frac{(1+\epsilon)\lambda c_2}{c_1} i \triangleq \mu i$$

epochs the consumed load was more than Θ. By making c_1 sufficiently large and ϵ sufficiently small, we can guarantee that $\mu < 1$. In this case, the probability that, among the $i-1$ epochs, fewer than μi consumed at least Θ load can be bounded by

$$\begin{aligned}\mathbf{Pr}(\mathcal{B}_i \cap \mathcal{M} \,|\, \mathcal{C}_t) &\le \binom{i-1}{(1-\mu)i}\left(\frac{1}{n^c}\right)^{(1-\mu)i}\\ &\le \left(\frac{e(i-1)}{(1-\mu)i}\right)^{(1-\mu)i}\left(\frac{1}{n^c}\right)^{(1-\mu)i}\\ &= \left(n^{-c}\cdot\frac{e}{1-\mu}\cdot\frac{i-1}{i}\right)^{(1-\mu)i}\\ &\le n^{-(1-\mu)(c-1)i}.\end{aligned} \tag{5.26}$$

By combining (5.23), (5.25), and (5.26) we get that

$$\mathbf{Pr}(\mathcal{B}_i \,|\, \mathcal{C}_t) \le 2n^{-(1-\mu)(c-1)i}.$$

Finally, we estimate the probability that the load is above Θ at time t using (5.22). If we make c and c_1 sufficiently large, we get that $(1-\mu)(c-1) > 0$, so the sum converges and we get

$$\mathbf{Pr}(\mathcal{B} \,|\, \mathcal{C}_t) = \sum_{i=1}^{\lfloor t/T_E \rfloor} 2n^{-(1-\mu)(c-1)i} \le \sum_{i=1}^{\infty} 2n^{-(1-\mu)(c-1)i} \le 3n^{-(1-\mu)(c-1)}.$$

From Theorem 5.6 we have $\lim_{t\to\infty}\mathbf{Pr}(\mathcal{C}_t) = 1$, which gives the result. □

5.3.2. Waiting time. By Theorem 5.1, part 2, we get that if the load of the system is bounded by Θ at some particular time, then with probability at least $1 - n^{-c} \ge 1 - n^{-c'}$ all the load that was in the system at that time is consumed during the next T_E steps. The limiting probability that the load of the system is above Θ is bounded by $3n^{-c'}$. Summing the failure probabilities proves the following.

THEOREM 5.11. *Consider a system whose load is injected by a strongly bounded adversary. Let W_t be the wait of a job that arrived at time t.*

$$\liminf_{t\to\infty} \mathbf{Pr}(W_t \le c_2\gamma^{-1}(w + \ln n)) \ge 1 - 4n^{-c'}.$$

The above holds without the limit if the system starts with no load.

6. Work stealing and the generator models. In this section we compare our results to the stochastic analysis of the *work stealing* load balancing protocol in [4]. In the work stealing protocol, nodes initiate load balancing steps only when their queues are empty. Concretely, after the new load generation, and before the load consumption, if a node u is empty, it selects a random neighbor, say v. If the load of v is $\ell_t(v)$, then v transfers half of its load to u, so that eventually u has load $\lfloor \ell_t(v) \rfloor$, while v ends up with load $\lceil \ell_t(v) \rceil$. Finally the nonempty nodes execute one job from their queue. The advantage of the work stealing protocol is that the total load balancing work in the network at any given time is proportional to the number of processors with empty queues. In particular, as long as all the processors are working, the network does not perform any load balancing. However, in this section we show that the work stealing protocol is either unstable, or stable with expected load exponential in n/d, where d is the minimum degree of the network. In contrast, protocol $\mathcal{P}$ is stable with expected load polynomial in n for any connected network topology.

We prove the gap in performance between the two protocols in the job-generator model introduced in [4]. We distinguish between two versions of this input model. In the deterministic generator model, an adversary places λn job-generators in the processors in an arbitrary fashion at the beginning of every step (having full information of the history of the system), and each generator adds one new job to the processor on which it is placed. In the stochastic job-generator model an adversary places n generators in the processors in an arbitrary fashion at the beginning of every time step (again having full information of the history), and each generator adds one new job to the processor on which it is placed with probability at most $\lambda < 1$, independently of the other generators.

Notice that both of these adversaries are special cases of a strongly bounded (λ, w, p, M) adversary, with $w = 1$, $p = 3$ and corresponding M being 0 for the deterministic and a constant for the stochastic. The deterministic model is clearly strongly bounded, while a standard Chernoff bound establishes the claim for the stochastic model. Then by Theorems 2.2 and 2.3, we get that, in both cases, the load and waiting time are bounded (both in expectation and with high probability) by $O(\gamma^{-1} n \ln n)$ and $O(\gamma^{-1} \ln n)$, respectively.

We contrast these results with the performance of the work stealing load balancing protocol on similar input. Consider first a deterministic job-generator adversary that adds, in each step, up to λn tokens to the n-node network. Let v be a node in the network with degree $d = o(n)$ (the claim is trivial for $d = \Omega(n)$, since then $n/d = O(1)$). In each step, the adversary puts $\lambda n - d$ tokens in v and one token in each of its neighbors. Since the queues of the d neighbors of v are never empty, v never participates in a load balancing step, and so its load increases (unboundedly) by $\lambda n - d - 1$ in each step.

Consider now the stochastic job-generator adversarial input process. Assume again a node v with minimal degree $d = o(n)$, and assume that in each step the adversary places exactly $m = \lfloor n/(d+1) \rfloor$ generators at v and at least m generators at each of its neighbors. Let $\ell_t(v)$ denote the load at v at time t.

Define $p = (2 + \lambda m/(1-\lambda))e^{-\lambda m}$ and assume that n is large, so that $p < 1$ (we use this fact later). Fix some time step $t \geq 1$. If node v's load did not increase during this time step, then either v did not produce many jobs (at most 1) or one of v's neighbors (say u_i, with degree $d_{u_i} \geq d$) was empty and chose to try to steal work from v, so v gave away load. For the latter to happen, u_i must have had no load (and in particular must have not produced any load at time t) and must have randomly chosen v out of its d_{u_i} neighbors. Thus,

$$\mathbf{Pr}(\ell_t(v) - \ell_{t-1}(v) < 1) \leq (1-\lambda)^m + \lambda m (1-\lambda)^{m-1} + \sum_{i=1}^{d} \frac{1}{d_{u_i}} (1-\lambda)^m \tag{6.1}$$
$$\leq p,$$

conditioned on all past events.

We now construct a random sequence $\{\ell'_t(v)\}_{t \geq 0}$, and we show that it is stochastically dominated by the sequence $\{\ell_t(v)\}$, as follows. We have $\ell'_0(v) = 0$, and for $t > 0$,

$$\ell'_t(v) = \begin{cases} \ell'_{t-1}(v)/2 & \text{with probability } p, \\ \ell'_{t-1}(v) + 1 & \text{with probability } 1-p, \end{cases}$$

with all the random choices being independent. Notice that by relation (6.1) the load of node v decreases with probability at most p. Moreover, if the load of node v decreases, the new load will be at least half of the old load, and hence each load $\ell_t(v)$ stochastically dominates the corresponding $\ell'_t(v)$. Therefore we have $\mathbf{E}[\ell_t(v)] \geq \mathbf{E}[\ell'_t(v)]$.

Using the recursion

$$\mathbf{E}[\ell'_t(v)] = \frac{p}{2}\mathbf{E}[\ell'_{t-1}(v)] + (1-p)(\mathbf{E}[\ell'_{t-1}(v)] + 1),$$

we compute

$$\mathbf{E}[\ell'_t(v)] = \frac{2(1-p)}{p}\left[1 - \left(1 - \frac{p}{2}\right)^t\right]$$

and

$$\begin{aligned}\mathbf{E}[L_t] &\geq \mathbf{E}[\ell_t(v)]\\ &\geq \mathbf{E}[\ell'_t(v)]\\ &= \frac{2(1-p)}{p}\left[1 - \left(1 - \frac{p}{2}\right)^t\right].\end{aligned}$$

Using the fact that $p < 1$, we get

$$\begin{aligned}\liminf_{t\to\infty} \mathbf{E}[L_t] &\geq \frac{2(1-p)}{p}\\ &= 2\left(\frac{1}{p} - 1\right)\\ &= \frac{2e^{\lambda\left\lfloor\frac{n}{d+1}\right\rfloor}}{2 + \frac{\lambda}{1-\lambda}\left\lfloor\frac{n}{d+1}\right\rfloor} - 2.\end{aligned}$$

Thus in the limit the expected total load in the system is at least exponential in n/d. In particular, if the network's minimum degree is constant, the expected total load in the system is exponential in the size of the network!

7. Conclusion.

We analyze a simple load balancing system and show that it has many desirable steady state properties under a big variety of input conditions. In particular, we derive low-degree polynomial bounds on the asymptotic expected load and waiting times of jobs in the system, and we match these expected performance results with high-probability results in a very natural restriction of the general stochastic adversary model. While there are many stability results for similar systems in the literature, our analysis of waiting time, the strength of our bounds, and our high-probability results are novel. In addition, our application of the Pemantle–Rosenthal result reveals many of the challenges in using general adversaries and will likely provide good insight for other applications. Finally, unlike much of the related work, our results hold for arbitrary connected network topologies and are optimal for the extremely important expander topology.

Plenty of open problems remain. In particular, our analysis cannot be easily modified to handle the case $\lambda = 1$, since we do not count the load that is consumed during the distribution phase of an epoch. Thus that case remains open. In addition, it is unknown whether Markovian adversaries are weaker than the general adversaries that we analyze. A result along these lines would have important ramifications in future analyses of network processes, since it would answer the question of whether Markov chain techniques can fill the role of the more general Pemantle and Rosenthal stochastic process results. And, of course, any results in an extension of our model allowing for asynchronous communication between nodes, nonuniform job execution time, and/or a continuous timescale would be very interesting.

Acknowledgments. We would like to thank the anonymous referees, whose elaborate comments significantly improved the overall presentation of the paper.

REFERENCES

[1] W. Aiello, B. Awerbuch, B. M. Maggs, and S. Rao, *Approximate load balancing on dynamic and asynchronous networks*, in Proceedings of the 25th Annual ACM Symposium on Theory of Computing (STOC'93), 1993, pp. 632–641.

[2] A. Anagnostopoulos, A. Kirsch, and E. Upfal, *Stability and efficiency of a random local load balancing protocol*, in Proceedings of the 44th IEEE Symposium on Foundations of Computer Science (FOCS'03), 2003, pp. 472–481.

[3] E. Anshelevich, D. Kempe, and J. Kleinberg, *Stability of load balancing in dynamic adversarial systems*, in Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC'02), 2002, pp. 399–406.

[4] P. Berenbrink, T. Friedetzky, and L. A. Goldberg, *The natural work-stealing algorithm is stable*, SIAM J. Comput., 32 (2003), pp. 1260–1279.

[5] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, and D. P. Williamson, *Adversarial queuing theory*, J. ACM, 48 (2001), pp. 13–38.

[6] W. Feller, *An Introduction to Probability Theory and Its Application, Vol.* 2, 2nd ed., John Wiley and Sons, New York, 1971.

[7] B. Ghosh, F. T. Leighton, B. M. Maggs, S. Muthukrishnan, C. G. Plaxton, R. Rajaraman, A. W. Richa, R. E. Tarjan, and D. Zuckerman, *Tight analyses of two local load balancing algorithms*, SIAM J. Comput., 29 (1999), pp. 29–64.

[8] B. Ghosh and S. Muthukrishnan, *Dynamic load balancing by random matchings*, J. Comput. System Sci., 53 (1996), pp. 357–370.

[9] S. Karlin and H. M. Taylor, *A First Course in Stochastic Processes*, 2nd ed., Academic Press, New York, 1975.

[10] R. Lüling and B. Monien, *A dynamic distributed load balancing algorithm with provable good performance*, in Proceedings of the 5th ACM Symposium on Parallel Algorithms and Architectures (SPAA'93), 1993, pp. 164–172.

[11] F. Meyer auf der Heide, B. Oesterdiekhoff, and R. Wanka, *Strongly adaptive token distribution*, in Proceedings of the 20th International Colloquium on Automata, Languages and Programming (ICALP'93), 1993, pp. 398–409.

[12] S. P. Meyn and R. L. Tweedie, *Markov Chains and Stochastic Stability*, Comm. Control Engrg. Ser., Springer-Verlag, London, New York, 1993.

[13] M. Mitzenmacher, *Load balancing and density dependent jump Markov processes*, in Proceedings of 37th IEEE Conference on Foundations of Computer Science (FOCS'96), 1996, pp. 213–222.

[14] M. Mitzenmacher, *Analyses of load stealing models based on differential equations*, in Proceedings of the 10th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA'98), 1998, pp. 212–221.

[15] S. Muthukrishnan and R. Rajaraman, *An adversarial model for distributed dynamic load balancing*, in Proceedings of the 10th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA'98), 1998, pp. 47–54.

[16] D. Peleg and E. Upfal, *The token distribution problem*, SIAM J. Comput., 18 (1989), pp. 229–243.

[17] R. Pemantle and J. S. Rosenthal, *Moment conditions for a sequence with negative drift to be uniformly bounded in L^r*, Stochastic Process. Appl., 82 (1999), pp. 143–155.

[18] Y. Rabani, A. Sinclair, and R. Wanka, *Local divergence of Markov chains and the analysis of iterative load balancing schemes*, in Proceedings of the 39th IEEE Symposium on Foundations of Computer Science (FOCS'98), 1998, pp. 694–705.

[19] L. Rudolph, M. Slivkin-Allalouf, and E. Upfal, *A simple load balancing scheme for task allocation in parallel machines*, in Proceedings of the 3rd Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA'91), 1991, pp. 237–245.

[20] C.-Z. Xu and F. C. M. Lau, *Iterative dynamic load balancing in multicomputers*, J. Oper. Res. Soc., 45 (1994), pp. 786–796.

[21] W. Zhu, C. Steketee, and B. Muilwijk, *Load balancing and workstation autonomy on Amoeba*, Aust. Comput. Sci. Commun., 17 (1995), pp. 588–597.