Tight Analyses of Two Local Load Balancing Algorithms

Bhaskar Ghosh [1] *F. T. Leighton* [2] *Bruce M. Maggs* [3,4]

S. Muthukrishnan [5] *C. Greg Plaxton* [6,7] *R. Rajaraman* [6,7]

Andréa W. Richa [3] *Robert E. Tarjan* [8] *David Zuckerman* [6,9]

August 3, 1995

CMU-CS-95-131

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

[1]Department of Computer Science, Yale University, New Haven, CT 06520. Supported by ONR Grant 4-91-J-1576 and a Yale/IBM joint study. Email: `ghosh@cs.yale.edu`.

[2]Department of Mathematics and Laboratory for Computer Science, MIT, Cambridge, MA 02139. Supported by ARPA Contracts N00014-91-J-1698 and N00014-92-J-1799. Email: `ftl@math.mit.edu`.

[3]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213. Email: `{aricha, bmm}@cs.cmu.edu`.

[4]Supported in part by an NSF National Young Investigator Award, No. CCR-9457766, and by ARPA Contract F33615-93-1-1330.

[5]DIMACS, Rutgers University, Piscataway, NJ 08855. Supported by DIMACS, Center for Discrete Mathematics and Theoretical Computer Science, a National Science Foundation Science and Technology Center, under NSF Contract STC-8809648. Email: `muthu@dimacs.rutgers.edu`.

[6]Department of Computer Science, University of Texas at Austin, Austin, TX 78712.
Email: `{diz, plaxton, rraj}@cs.utexas.edu`.

[7]Supported by the Texas Advanced Research Program under Grant No. ARP-93-003658-461.

[8]Department of Computer Science, Princeton University, 35 Olden Street, Princeton, NJ 08544, and NEC Research Institute. Research at Princeton University supported in part by the National Science Foundation, Grant No. CCR-8920505, and the Office of Naval Reserach, Contract No. N0014-91-J-1463. Email: `ret@cs.princeton.edu`.

[9]Supported in part by an NSF National Young Investigator Award, No. CCR-9457799.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of any of the sponsoring agencies or the U.S. Government.

Abstract

This paper presents an analysis of the following load balancing algorithm. At each step, each node in a network examines the number of tokens at each of its neighbors and sends a token to each neighbor with at least $2d+1$ fewer tokens, where d is the maximum degree of any node in the network. We show that within $O(\Delta/\alpha)$ steps, the algorithm reduces the maximum difference in tokens between any two nodes to at most $O((d^2 \log n)/\alpha)$, where Δ is the maximum difference between the number tokens at any node initially and the average number of tokens, n is the number of nodes in the network, and α is the edge expansion of the network. The time bound is tight in the sense that for any graph with edge expansion α, and for any value Δ, there exists an initial distribution of tokens with imbalance Δ for which the time to reduce the imbalance to even $\Delta/2$ is at least $\Omega(\Delta/\alpha)$. The bound on the final imbalance is tight in the sense that there exists a class of networks that can be locally balanced everywhere (i.e., the maximum difference in tokens between any two neighbors is at most $2d$), while the global imbalance remains $\Omega((d^2 \log n)/\alpha)$. Furthermore, we show that upon reaching a state with a global imbalance of $O((d^2 \log n)/\alpha)$, the time for this algorithm to locally balance the network can be as large as $\Omega(n^{1/2})$. We extend our analysis to a variant of this algorithm for dynamic and asynchronous networks. We also present tight bounds for a randomized algorithm in which each node sends at most one token in each step.

Keywords: load balancing, distributed network algorithms.

1 Introduction

A natural way to balance the workload in a distributed system is to have each work station periodically poll the other stations to which it is connected, and send some of its work to stations with less work pending. This paper analyzes the effectiveness of this local load balancing strategy in the simplified scenario in which each work station has a collection of independent unit-size jobs (called *tokens*) that can be executed on any other work station. We model a distributed system as a graph, where nodes correspond to work stations, and edges correspond to connections between stations, and we assume that in one unit of time, at most one token can be transmitted across an edge of the graph in each direction. Our analysis addresses only the static load balancing aspect of this problem; we assume that each processor has an initial collection of tokens, and that no tokens are created or destroyed while the tokens are being balanced.

We analyze the algorithms in this paper in terms of the initial *imbalance* of tokens, i.e., the maximum difference between the number of tokens at any node and the average number of tokens, which we denote Δ, the number of nodes in the graph, which we denote n, the maximum degree of the graph, d, and the node and edge expansion of the graph. We define the *node expansion* μ, of a graph G to be the largest value such that every set S of $n/2$ or fewer nodes in G has at least $\mu|S|$ neighbors outside of S. We define the *edge expansion* α, of a graph G to be the largest value such that for every set S of $n/2$ or fewer nodes in G, there are at least $\alpha|S|$ edges in G with one endpoint in S and the other not in S.

The performance of an algorithm is characterized by the time that it takes to balance the tokens, and by the final balance that it achieves. We say that an algorithm *globally balances* (or just *balances*) to within t tokens if the maximum difference in the number of tokens between any two nodes in the graph is at most t. We say that an algorithm *locally balances* to within t tokens if the maximum difference in the number of tokens between any neighboring nodes in the graph is at most t.

We analyze two different types of algorithms in this paper, single-port and multi-port. In the *single-port* model, a node may transmit or receive at most one token in one unit of time. In the *multi-port* model, a node may simultaneously transmit or receive a token across all of its edges (there may be as many as d) in a single unit of time. Not surprisingly, the load balancing algorithms run faster in the multi-port model. In practice, however, single-port nodes may be prefered to multi-port nodes because they are easier and less costly to build.

1.1 Our results

This paper analyzes the simplest and most natural local algorithms in both the single-port and multi-port models.

In the single-port algorithm, a matching is randomly chosen at each step. First, each (undirected) edge in the network is independently selected to be a *candidate* with probability $1/4d$. Then each candidate edge (u, v) for which there is another candidate edge (u, x) or (y, v) is removed from the set of candidates. The remaining candidates form a matching M in the graph. For each edge (u, v) in M, if u and v have the same number of tokens, then nothing is sent across (u, v). Otherwise, a token is sent from the node with more tokens to the node with fewer. This algorithm was first analyzed in [14].

We analyze the performance of the single-port algorithm in terms of both the edge expansion and the node expansion of the graph. In terms of edge expansion, we show that the single-port algorithm balances to within $O(d \log n/\alpha)$ tokens in $O(d\Delta/\alpha)$ steps, with high probability. In terms of node expansion, the final imbalance is $O(\log n/\mu)$, and the time is $O(d\Delta/\mu)$, with high probability. (To compare these bounds, note that $\mu \leq \alpha \leq d\mu$.) The time bounds are tight in the sense that for many values of n, d, α, and Δ, there is an n-node maximum degree d graph with edge expansion α or node expansion μ and an initial placement of tokens with imbalance Δ where the time (for any algorithm) to balance to within even $\Delta/2$ tokens is at least $\Omega(d\Delta/\alpha)$. Similarly, in terms of node expansion, there exist classes of graphs where the time to balance to within even $\Delta/2$ tokens is at least $\Omega(d\Delta/\mu)$.

The multi-port algorithm is simpler and deterministic. At each step, a token is sent from node u to node v across edge (u, v) if at the beginning of the step node u contained at least $2d + 1$ more tokens than node v. This algorithm was first analyzed in [2].

As in the single-port case, we analyze the multi-port algorithm in terms of both edge expansion and node expansion. In terms of edge expansion, the algorithm balances to within $O(d^2 \log n/\alpha)$ tokens in $O(\Delta/\alpha)$ steps. This bound is tight in the sense that for any network with edge expansion α, and any value Δ, there exists an initial distribution of tokens with imbalance Δ such that the time to reduce the imbalance to even $\Delta/2$ is $\Omega(\Delta/\alpha)$. In terms of node expansion, the algorithm balances to within $O(d \log n/\mu)$ tokens in $O(\Delta/\mu)$ time. This bound is tight in the sense that for many values of d, n, and μ, and any value Δ, there exists an n-node, maximum degree d graph with node expansion μ and an initial distribution of tokens with imbalance Δ for which the time to balance to within $\Delta/2$ tokens is $\Omega(\Delta/\mu)$.

Both the single-port and multi-port algorithms will eventually locally balance the network, the single-port algorithm to within one token, and the multi-port algorithm to within $2d$ tokens. However, even after reducing the global imbalance to a small value, the time for either of these algorithms to reach a locally balanced state can be quite large. In particular, we show that after reaching a state that is globally balanced to within $O(d \log n/\mu)$ tokens, the multi-port algorithm may take another $\Omega(n^{1/2})$ steps to reach a state that is locally balanced to within $2d$ tokens. For networks with large node expansion and small degree, e.g., $\mu = \Omega(1)$ and $d = O(1)$, and small initial imbalance, e.g., $\Delta = O(d \log^2 n/\mu)$, the time to locally balance the network, $\Omega(n^{1/2})$, may be much larger than the time, $O(\Delta/\mu) = O(d \log^2 n/\mu^2) = O(\log^2 n)$ to reach a state that is globally balanced to within $O(d \log n/\mu)$ tokens. We prove similar bounds in terms of edge expansion and also for the single-port algorithm.

Thus far we have described a network model in which the nodes are synchronized by a global clock (i.e., a *synchronous* network), and in which the edges are assumed not to fail. With minor modifications, however, the load balancing algorithms can be made to work in both asynchronous and dynamic networks. In a *dynamic* network, the set of edges in the network may vary at each time step. In any time step, a *live* edge is one that can transmit one message in each direction. We assume that at each time step, each node in a synchronous dynamic network knows which of its edges are live. In an *asynchronous* network, the topology is fixed, but an adversary determines the speed at which each edge operates at every instant of time. For every undirected edge between two nodes, we allow at most two messages to be in transit at any instant in time. These messages may travel in opposite directions across the edge, or both may travel in one direction, while no message travels in the opposite direction. An edge is said to be *live* for a unit interval of time if every message that was in transit across the edge (in either direction) at the beginning of the interval is guaranteed to reach the end of the edge by the end of the interval. We analyze the performance of the multi-port load balancing algorithm under the assumption that at each time step, the set of live edges has some edge expansion α, or node expansion μ.

We also study the off-line load balancing problem, in which every node has knowledge of the global state of the network. This problem on has been studied on static synchronous networks in [28]. We use their results to obtain tight bounds on off-line load balancing in terms of edge expansion and node expansion. In particular, we prove that any network can be balanced off-line in $\lceil(1+\mu)\Delta/\mu\rceil$ steps such that no node has more than two tokens over the average. This result can be used to show that any network can be balanced off-line to within three tokens in at most $2\lceil(1+\mu)\Delta/\mu\rceil$ steps in the single-port model. Moreover, there exists a network and an initial token distribution for which any single-port off-line algorithm takes more than $\lceil(1+\mu)\Delta/\mu\rceil$ steps to balance the network to within one token. Similarly, in the multi-port model, any network can be balanced off-line in at most $\lceil\Delta/\alpha\rceil$ steps so that no node contains more than d tokens over the average. Using this result, we show that any network can be balanced to within $d+1$ tokens in at most $2\lceil\Delta/\alpha\rceil$ steps. It is easy to observe that for any network G there exists an initial token distribution such that any algorithm will take at least $\lceil\Delta/\alpha\rceil$ steps to balance G to within one token.

1.2 Previous and related work

Load balancing has been studied extensively since it comes up in a wide variety of settings including adaptive mesh partitioning [16, 38], fine grain functional programming [15], job scheduling in operating systems [13, 24], and distributed game tree searching [21, 25]. A number of models have been proposed for load balancing, differing chiefly in the amount of global information used by the algorithm [2, 11, 12, 14, 26, 30]. On these models, algorithms have been proposed for specific applications; also, proposed heuristics and algorithms have been analyzed using simulations and queuing–theoretic techniques [27, 34, 36]. In what follows, we focus on models that allow only local algorithms and on prior work that takes an analytical approach to the load balancing problem.

Local algorithms restricted to particular networks have been studied on counting networks [4, 22], hypercubes [19, 33], and meshes [16, 28]. Another class of networks on which load balancing has been studied is the class of expanders. Peleg and Upfal [31] pioneered this study by identifying certain small-degree expanders as being suitable for load balancing. Their work has been extended in [9, 17, 32]. These algorithms either use strong expanders to approximately balance the network, or the AKS sorting network [3] to perfectly balance the network. Thus, they do not work on networks of arbitrary topology. Also, these algorithms work by setting up fixed paths through the network on which load is moved and therefore fail when the network changes. In contrast, our local algorithm works on any arbitrary dynamic network that remains connected.

On arbitrary topologies, load balancing has been studied under two models. In the first model, any amount of load can be moved across a link in any time step [8, 12, 14, 18, 35]. The second model is the one that we adopt here, namely one in which at most one unit load can be moved across a link in each time step. Load balancing algorithms on the second model were first proposed and analyzed in [2] for the multi-port variant and in [14] for the single-port variant. The upper bounds established by them are suboptimal by a factor of $\Omega(\sqrt{n})$ or $\Omega(\log(n\Delta))$ in general. Our result here is an improved, in fact, an optimal bound for the same problem.

As remarked earlier, our multi-port results (and those in [2]) hold even for dynamic or asynchronous networks. In general, work on dynamic and asynchronous networks has been limited. In work related to load balancing for instance, an end-to-end communication problem, namely one in which messages are routed from a single source to a single destination, has been studied in [1, 7] on dynamic networks. Our scenario is substantially more involved since we are required to move load between several sources and destinations simultaneously. Another result on dynamic networks is the recent analysis of a local algorithm for the approximate multicommodity flow problem [5, 6]. While their result has several applications including the end-to-end communication problem mentioned above, it does not seem to extend to load balancing. Our result on load balancing is related to their work in the technique; however, our algorithm and analysis are simpler and we obtain optimal bounds for our problem.

The convergence of local load balancing algorithms is related to that of random walks on Markov Chains. Indeed the convergence bounds in both cases depend on the expansion properties of the underlying graph and they are established using potential function arguments. There are however two important differences. First, the analysis of the rapid convergence of random walks [20, 29] relies on averaging arbitrary probabilities across any edge. This corresponds to sending an arbitrary (possibly nonintegral) load along an edge which is forbidden in our model. In this sense, the analysis in [12] (and all references in the unbounded capacity model) are similar to the random walk analysis. Second, our argument uses an exponential potential function. The analyses in [12, 20, 29], in contrast, use quadratic potential functions. Our potential function and our amortized analysis were necessary since a number of previous attempts using quadratic potential functions yielded suboptimal results [2, 14] for local load balancing.

1.3 Outline

The remainder of this paper is organized as follows. Section 2 contains some definitions. Section 3.1 analyzes the performance of the single-port algorithm. Section 3.2 analyzes the performance of the multi-port algorithm. In Section 4, we show that the time to reach a locally balanced state can be quite large, even if the network starts in a state that is well balanced globally. Section 5 describes extensions to dynamic and asynchronous networks. Finally, Section 6 presents tight bounds on off-line load balancing.

2 Preliminaries

For any network $G = (V, E)$ with n nodes, m edges, and edge expansion α, we denote the number of tokens at $v \in V$ by $w(v)$. We denote the average number of tokens by ρ. For simplicity, throughout this paper we assume that ρ is an integer. We assign a unique *rank* from $[1, w(v)]$ to every token at v. The *height* of a token is its rank minus ρ. The height of a node is the maximum among the heights of all its tokens.

Consider a partition of V given by $\{S_i\}$, where the index i is an integer and S_i may be empty for any i. Let $S_{>j}$ be $\cup_{i>j} S_i$. Similarly we define $S_{\geq j}$, $S_{<j}$, and $S_{\leq j}$. We define index i to be *good* if $|S_i| \leq \alpha |S_{>i}|/2d$. An index that is not good is called a *bad* index. For any bad index i, we have $|S_{>i}| < |S_{\geq i}|/(1 + \alpha/(2d))$. Since $|S_{>0}|/(1+\alpha/(2d))^{\log_{(1+\alpha/(2d))} n} \leq 1$, there can be at most $\lceil \log_{(1+\alpha/(2d))} n \rceil$ bad indices. It follows that at least half of the indices in $[1, 2\lceil \log_{(1+\alpha/(2d))} n \rceil]$ are good.

3 Analysis for static synchronous networks

3.1 The single-port model

In this section, we analyze the single-port load balancing algorithm that is described in Section 1.1.

Theorem 3.1 *For an arbitrary network G with n nodes, maximum degree d, edge expansion α, and initial imbalance Δ, the single-port algorithm balances within $O((d \log n)/\alpha)$ tokens in $O((d\Delta)/\alpha)$ steps, with high probability.*

Before every step we partition the set of nodes according to how many tokens they contain. For every integer i, we denote the set of nodes having $\rho + i$ tokens as S_i. Consider the first T steps of the algorithm, with T to be specified later. It holds that either $|S_{>0}| \leq n/2$ at the start of at least half the steps, or $|S_{\leq 0}| \leq n/2$ at the start of at least half the steps. Without loss of generality, assume the former is true. Since at least half of the indices in $[1, 2\lceil \log_{(1+\alpha/(2d))} n \rceil]$ are good in any time step t, there exists an index j in $[1, 2\lceil \log_{(1+\alpha/(2d))} n \rceil]$ that is good in at least half of those time steps in which $|S_{>0}| \leq n/2$. Hence j is good in at least $T/4$ steps.

With every token at height h we associate a potential of $\phi(h)$, where $\phi : N \rightarrow R$ is defined as follows:

$$\phi(x) = \begin{cases} 0 & \text{if } x \leq j, \\ (1+\nu)^x & \text{otherwise,} \end{cases} \tag{1}$$

where $\nu = \alpha/(cd)$, and $c > 1$ is a real constant to be specified later. The potential of the network is the sum of the potentials of all tokens in the network. While transmitting a token, every node sends its token with maximum height. Similarly, any token arriving into a node with height h is assigned height $h + 1$. It follows from the definition of the potential function, and the fact that the height of a token never increases, that the potential of the network never increases. In the following, we show that during any step when j is good, the expected decrease in potential of the network is at least an $\varepsilon\nu^2$ fraction of the potential before the step, where $\varepsilon > 0$ is a real constant to be specified later.

Before proving Theorem 3.1, we present an informal outline of the proof. For simplicity, let us assume that G is a constant-degree expander, i.e., $d = O(1)$ and $\mu = \Omega(1)$. Consider the scenario in which all of the indices greater than j are bad. In this situation, for every i greater than j, the size of the set $S_{\geq i}$ grows exponentially with decreasing i, and hence the number of tokens with height i grows exponentially with decreasing i. If the growth of $\phi(x)$ with increasing x is slower than the growth of $S_{\geq i}$ with decreasing i, then the total potential due to tokens at height i "dominates" the total potential due to tokens at height greater than i. In such a case the potential of $S_{>j}$ is essentially a constant times the potential of tokens at height $j+1$. In addition, if the potential of tokens of height at most j is zero, then in every step when j is good, there is a constant fraction potential drop, because a constant fraction of nodes in $S_{>j}$ send tokens to $S_{<j}$ in such a step. The exponential function we have defined in Equation (1) satisfies the properties described above.

In general, the indices greater than j could form any sequence of good and bad indices that respects the upper bound on the number of bad indices. We consider the indices greater than j in reverse order and show by an amortized analysis that for each index i we can "view" all indices greater than or equal to i as bad. If i is bad, then this view is trivially preserved; otherwise, there is a significant potential drop across the cut $(S_{\leq i}, S_{>i})$, and this drop can be used to rearrange the potential of $S_{>i}$ in order to maintain the view that all indices greater than i are bad. We then invoke the argument for the case in which all indices greater than j are bad and complete the proof.

Consider step t of the algorithm. Let Φ_t denote the potential of the network after step $t > 0$. Let M_i be the set of tokens that are sent from a node in $S_{>i}$ to a node in $S_{<i}$. Let $m_i = |M_i|$. We say that a token p has an *i-drop* of $\phi(i+1) - \phi(i)$, if p moves from a node in $S_{>i}$ to a node in $S_{<i}$. Thus, the potential drop due to a token moving on an edge from node $u \in S_i$ to node $v \in S_{i'}$, $i > i'+1$, can be expressed as the sum of k-drops for $i' < k < i$. In Lemma 3.1, we use this notion of i-drops to relate the total potential drop in step t, Ψ, to the $m_i's$.

Lemma 3.1

$$\Psi = \left(\sum_{i>j} m_i \nu (1+\nu)^i\right) + m_j (1+\nu)^{j+1}.$$

Proof: Let M be the set of tokens that are moved from a node in $S_{>j}$. (Note that tokens that start from and end at nodes in $S_{>j}$ also belong to M.) For any token p, let $a(p)$ (resp., $b(p)$) be the height of p before (resp., after) step t.

$$\begin{aligned}
\Psi &= \sum_{p\in M} (\phi(a(p)) - \phi(b(p))) \\
&= \sum_{p\in M, p\notin M_j} \sum_{b(p)\leq i<a(p)} (\phi(i+1) - \phi(i)) + \sum_{p\in M_j} \sum_{j\leq i<a(p)} (\phi(i+1) - \phi(i)) \\
&= \left(\sum_{i>j} \sum_{p\in M_i} \nu(1+\nu)^i\right) + \left(\sum_{p\in M_j} (1+\nu)^{j+1}\right) \\
&= \left(\sum_{i>j} m_i \nu (1+\nu)^i\right) + m_j (1+\nu)^{j+1}.
\end{aligned}$$

(In the third step we rearrange the terms in the summations and also use the equation $\phi(i+1) - \phi(i) = \nu(1+\nu)^i$ for all $i > j$.) ■

We now describe the amortized analysis, alluded to earlier in this section, that we use to prove Theorem 3.1. We associate a charge of $\varepsilon\nu^2\phi(h)$ with each token at height h. We show that we can pay for all the charges using the expected potential drop $E[\Psi]$, and thus place a lower bound on $E[\Psi]$. We consider the indices in $[j+1, \ell]$ in reverse order, where ℓ is the maximum token height. Corresponding to every i in $[j, \ell]$, we maintain a "debt" term, given by Γ_i below, which is the difference between the charges due to tokens at height greater than i and the sum of i'-drops for $i' > i$. Hence Γ_i is calculated by subtracting the sum of i-drop's from Γ_{i+1} and adding the charges due to tokens at height i. We will show that $E[\Gamma_i]$ is such that the indices in $[i+1, \ell]$ can be viewed as a sequence of bad indices. In other words, we upper bound $E[\Gamma_i]$ by $O(\nu|S_{\geq i}|(1+\nu)^i)$. It follows from this upper bound and the informal argument outlined earlier in this section that $E[\Gamma_{j+1}]$, i.e., the expected total debt, can be paid for by the expected drop across index j.

Formally, for any $i > j$, we define

$$\begin{aligned}
\Psi_i &= \sum_{k \geq i} m_k \nu (1+\nu)^k, \text{ and} \\
\Gamma_i &= (\varepsilon\nu^2)\left(\sum_{p:a(p)\geq i} (1+\nu)^{a(p)}\right) - \Psi_i.
\end{aligned}$$

We also define

$$\Gamma = (\varepsilon\nu^2)\left(\sum_{p:a(p)>j} (1+\nu)^{a(p)}\right) - \Psi.$$

Note that $\Phi_{t-1} = \sum_{p:a(p)>j} (1+\nu)^{a(p)}$ is the total potential of $S_{>j}$ prior to step t.

In order to prove the upper bound on $E[\Gamma_i]$, we place a lower bound on $E[m_i]$ that is obtained from the following lemma of [14].

Lemma 3.2 ([14]) *For any edge $e \in E$, the probability that e is selected in the matching is at least $1/(8d)$.* ■

Lemma 3.3 *There exists a real constant $\varepsilon > 0$ such that for all $i > j$, we have $E[\Gamma_i] \leq (\varepsilon\nu)|S_{\geq i}|(1+\nu)^i$.*

Proof: The proof is by reverse induction on i. For $i \geq \ell$, the claim holds trivially. Therefore, for the base case we consider $i = \ell$. Since $m_\ell = 0$, we have $\Psi_\ell = 0$. Thus, $\Gamma_\ell = (\varepsilon\nu^2)|S_\ell|(1+\nu)^\ell \leq (\varepsilon\nu)|S_{\geq\ell}|(1+\nu)^\ell$, since $\nu = \alpha/(cd) \leq 1/c \leq 1$ by our choice of c.

For the induction step we consider two cases, depending on whether i is good or bad. If i is good, then $|S_i| \leq \alpha|S_{>i}|/2d$. Since there are at most $\alpha|S_{>i}|/2$ edges from $S_{>i}$ to S_i, by the expansion property of the graph we find that there are at least $\alpha|S_{>i}|/2$ edges from $S_{>i}$ to $S_{<i}$. By Lemma 3.2 we have $E[m_i] \geq \alpha|S_{>i}|/(16d)$. Therefore, we have

$$\begin{aligned}
E[\Gamma_i] &= E[\Gamma_{i+1}] + (\varepsilon\nu^2)|S_{\geq i}|(1+\nu)^i - E[m_i]\nu(1+\nu)^i \\
&\leq E[\Gamma_{i+1}] + (\varepsilon\nu^2)|S_{\geq i}|(1+\nu)^i - c\nu^2|S_{>i}|(1+\nu)^i/16 \\
&\leq E[\Gamma_{i+1}] - (\nu^2)|S_{\geq i}|(1+\nu)^i(f(c,\alpha,d) - \varepsilon) \\
&\leq (\varepsilon\nu)|S_{>i}|(1+\nu)^{i+1} - (\nu^2)|S_{\geq i}|(1+\nu)^i(f(c,\alpha,d) - \varepsilon) \\
&\leq (\varepsilon\nu)|S_{\geq i}|(1+\nu)^i((1+\nu) - \nu(f(c,\alpha,d) - \varepsilon)/\varepsilon),
\end{aligned}$$

where $f(c,\alpha,d) = c/(16(1+\alpha/(2d)))$. (The third inequality follows from the fact that $|S_{>i}| \geq |S_{\geq i}|/(1+\alpha/(2d))$. The fourth inequality follows from the induction hypothesis.)

The second case is when i is bad. Thus $|S_i| \geq \alpha|S_{>i}|/(2d)$. We have

$$\begin{aligned}
E[\Gamma_i] &\leq E[\Gamma_{i+1}] + (\varepsilon\nu^2)|S_{\geq i}|(1+\nu)^i \\
&\leq (\varepsilon\nu)|S_{>i}|(1+\nu)^{i+1} + (\varepsilon\nu^2)|S_{\geq i}|(1+\nu)^i \\
&\leq (\varepsilon\nu)|S_{\geq i}|(1+\nu)^i((1+\nu)/(1+c\nu/2) + \nu).
\end{aligned}$$

We now complete the induction step by determining values for c and ε such that the inequalities

$$((1+\nu) - \nu(f(c,\alpha,d) - \varepsilon)/\varepsilon) \leq 1 \tag{2}$$

$$(1+\nu)/(1+c\nu/2) + \nu \leq 1 \tag{3}$$

hold, and thus establish the induction step. Since $\alpha \leq d$, we can set c to be any constant greater than or equal to $(\alpha/d)+4$ (e.g., $c=5$). For this choice of c, $\nu \leq (c-4)/c$, and hence $\nu \leq (c\nu - c\nu^2)/4$. Therefore,

$$\begin{aligned}(1+\nu)/(1+c\nu/2)+\nu &= (1+2\nu+c\nu^2/2)/(1+c\nu/2)\\ &\leq (1+c\nu/2)/(1+c\nu/2) = 1.\end{aligned}$$

Thus, Equation (3) is satisfied. Since $\alpha \leq d$, we find that $f(c,\alpha,d) \geq c/24$. We now set $\varepsilon = c/48$ to establish Equation (2). (For example, $c=5$ and $\varepsilon = 5/48$.) ■

Applying Lemma 3.3 with $i = j+1$, it follows that $E[\Gamma_{j+1}] \leq (\varepsilon\nu)|S_{\geq j+1}|(1+\nu)^{j+1}$. If j is good then $E[m_j] \geq \alpha|S_{>j}|/16d$, and by the definitions of Γ, Γ_{j+1}, and Ψ, we have

$$\begin{aligned}E[\Gamma] &= E[\Gamma_{j+1}] - E[m_j](1+\nu)^{j+1}\\ &\leq E[\Gamma_{j+1}] - \alpha|S_{>j}|(1+\nu)^{j+1}/(16d)\\ &\leq (\varepsilon\nu)|S_{>j}|(1+\nu)^{j+1} - \alpha|S_{>j}|(1+\nu)^{j+1}/(16d)\\ &\leq \nu|S_{>j}|(1+\nu)^{j+1}(\varepsilon - c/16)\\ &\leq 0,\end{aligned}$$

since $c/16 \geq \varepsilon$.

By the definitions of Ψ and Γ, we have $\Phi_t \leq \Phi_{t-1} - \Psi$ and $\Gamma = \varepsilon\nu^2\Phi_{t-1} - \Psi$. If j is good during step t, we have $E[\Gamma] \leq 0$, and therefore, $E[\Phi_t] \leq \Phi_{t-1}(1-\varepsilon\nu^2)$, where the expectation is taken over the random matching selected in step t. Thus, we have $E[\Phi_{t+T}] \leq \Phi_t(1-\varepsilon\nu^2)^{T/4}$, where the expectation is over all the random matchings in the T steps. By setting $T = \lceil (4\ln 4)/(\varepsilon\nu^2)\rceil$, we obtain $E[\Phi_{t+T}] \leq \Phi_t/4$. By Markov's inequality, the probability that $\Phi_{t+T} \geq \Phi_t/2$ is at most $1/2$. Therefore, using standard Chernoff bounds [10], we can show that in $T' = 8aT\lceil(\log\Phi_0 + \log n)\rceil$ steps, $\Phi_{T'} > 1$ with probability at most $O(1/(\Phi_0)^a + 1/n^a)$ for any constant $a > 0$. Since $\Phi_0 \leq n(1+\nu)^{\Delta+1}/\nu$, we have $\log\Phi_0 \leq (\Delta+1)(\nu) + \log n - \log\nu$. Therefore, for $T' = O(\Delta d/\alpha + d^2\log n/\alpha^2)$, we have $\Phi_t < 1$ with high probability which implies that after T' steps, $|S_{>2\log_{1+\alpha/(2d)} n}| = 0$ with high probability.

To establish balance in the number of tokens below the average, we use an averaging argument to show that after T' steps $|S_{<-2\log_{1+\alpha/(2d)} n}| \leq n/2$ with high probability, and then repeat the above arguments redefining the potential appropriately. This proves Theorem 3.1.

3.2 The multi-port model

In this section, we analyze the deterministic multi-port algorithm described in Section 1.1.

Theorem 3.2 *For an arbitrary network G with n nodes, maximum degree d, edge expansion α, and initial imbalance Δ, the multi-port algorithm load balances to within $O((d^2\log n)/\alpha)$ tokens in $O(\Delta/\alpha)$ steps.*

The proof of Theorem 3.2 is similar to that of Theorem 3.1. We assign a potential to every token, that grows exponentially with height. We then show by means of an amortized analysis that a suitable rearrangement of the potential reduces every instance of the problem to a special instance that we understand well.

Before every step we partition the set of nodes according to how many tokens they contain, where groups are separated by $2d$. For every integer i, we denote the set of nodes having between $\rho - d + 2id$ and $\rho + d - 1 + 2id$ tokens as S_i.

Consider the first T steps of the algorithm with T to be specified later. Without loss of generality, we assume that $|S_{>0}| \le n/2$ holds in at least half of these steps. By Section 2, there exists an index j in $[1, 2\lceil \log_{1+\alpha/(2d)} n \rceil]$ that is good in at least half of those steps in which $|S_{>0}| \le n/2$. Hence in T steps of the algorithm, j is good in at least $T/4$ steps.

With every token at height h we associate a potential of $\phi(h)$, where $\phi : N \rightarrow R$ is defined as follows:

$$\phi(x) = \begin{cases} 0 & \text{if } x \le 2jd, \\ (1+\nu)^x & \text{otherwise,} \end{cases}$$

where $\nu = \alpha/(cd^2)$, and $c > 0$ is a constant to be specified later. The potential of the network is the sum of the potentials of all tokens in the network. While transmitting some number (say m) of tokens in a particular step, a node sends the m highest-ranked tokens. Similarly if m tokens arrive at a node during a step, they are assigned the m highest ranks within the node. Thus, tokens that do not move retain their ranks after the step. It follows from the definition of the potential function and the fact that the height of a token never increases that the network potential never increases. In the following we show that whenever j is good the potential of $S_{>j}$ decreases by a factor of $\varepsilon\nu^2 d^2$, where $\varepsilon > 0$ is a real constant to be specified later. (For the sake of simplicity, we assume that d is even. If d is odd, we can replace d by $d+1$ in our argument without affecting the bounds up to constant factors.)

Consider step t of the algorithm. Let u be a node in $S_{<i}$ with height h. Let A (resp., B) be the set of tokens that u receives from nodes in $S_{>i}$ (resp., $S_{\le i}$). We assign new ranks to tokens in A and B such that the rank of every token in A is less than that of every token in B. Let C be the set of tokens in A that attain height at most $h + (d/2)$ after the step. Since $|A| \le d$, by the choice of our ranking, we have $|C| \ge |A|/2$. We call C the set of *primary* tokens. Also, all tokens leaving node u are at height at least $h - d + 1$ prior to this step.

For any token p, let $a(p)$ (resp., $b(p)$) be the index of the group of the node that contains p before (resp., after) the step. (Note that the indexing is done prior to the step.) Let M_i be the set of primary tokens received by nodes in $S_{<i}$. Let $m_i = |M_i|$. (Note that m_i is at least half the number of edges joining some node in $S_{<i}$ and some node in $S_{>i}$.) Lemma 3.4 establishes the relationship between the total potential drop Ψ in step t and the m_i's.

Lemma 3.4

$$\Psi \ge \left(\frac{1}{2} \sum_{i>j} m_i \nu d (1+\nu)^{(2i-1)d} \right) + m_j (1+\nu)^{2jd+1}.$$

Proof: Let M be the set of primary tokens that are moved from nodes in $S_{>j}$. (Note that primary tokens that start from a node in $S_{>j}$ and end at a node in $S_{>j}$ are in M.) By the definition of primary token, the height of p prior to the step is at least $2a(p)d - 2d + 1$ and the height after the step is at most $2b(p)d + 3d/2$. Moreover, p belongs to M_i for all i such that $b(p) < i < a(p)$.

$$\begin{aligned}
\Psi &\ge \sum_{p \in M} [\phi(2a(p)d - 2d + 1) - \phi(2b(p)d + 3d/2)] \\
&\ge \sum_{p \in M, p \notin M_j} \sum_{b(p) < i < a(p)} [\phi(2(i+1)d - 2d + 1) - \phi(2(i-1)d + 3d/2)] \\
&\quad + \sum_{p \in M_j} \left(\sum_{j < i < a(p)} [\phi(2(i+1)d - 2d + 1) - \phi(2(i-1)d + 3d/2)] + \phi(2(j+1)d - 2d + 1) \right) \\
&\ge \left(\frac{1}{2} \sum_{i>j} \sum_{p \in M_i} \nu d (1+\nu)^{2id-d} \right) + \sum_{p \in M_j} (1+\nu)^{2d(j+1)-2d+1}
\end{aligned}$$

$$\geq \left(\frac{1}{2}\sum_{i>j} m_i \nu d(1+\nu)^{2id-d}\right) + m_j(1+\nu)^{2jd+1}.$$

■

For any token p, let $h(p)$ denote the height of p prior to the step. Thus $2a(p)d - d \leq h(p) \leq 2a(p)d + d - 1$. For $i > j$ and for a suitable constant $\varepsilon > 0$ to be specified later, we define

$$\begin{aligned}\Psi_i &= \frac{1}{2}\sum_{k\geq i} m_k \nu d(1+\nu)^{2kd-d}, \text{ and}\\ \Gamma_i &= (\varepsilon\nu^2 d^2)\left(\sum_{p:h(p)\geq 2id-d}(1+\nu)^{h(p)}\right) - \Psi_i.\end{aligned}$$

We also define

$$\Gamma = (\varepsilon\nu^2 d^2)\left(\sum_{p:h(p)>2jd}(1+\nu)^{h(p)}\right) - \Psi.$$

For any step t', let $\Phi_{t'}$ denote the total potential after step t'. Thus, $\Phi_{t-1} = \sum_{p:h(p)>2jd}(1+\nu)^{h(p)}$ is the total potential prior to step t.

Lemma 3.5 *There exists a real constant $\delta > 0$ such that for all $i > j$, we have*

$$\Gamma_i \leq (\delta\nu d^2)|S_{\geq i}|(1+\nu)^{2id-d}.$$

Proof: The proof is by reverse induction on i. Let ℓ be the maximum token height. For $i > \lfloor(\ell+d)/2d\rfloor$, the claim is trivial. Therefore, for the base case we consider $i = \lfloor(\ell+d)/2d\rfloor$. Since $\Psi_i = 0$, we have

$$\begin{aligned}\Gamma_i &\leq (2\varepsilon\nu^2 d^3)|S_i|(1+\nu)^{\ell}\\ &\leq (2\varepsilon\nu^2 d^3)|S_{\geq i}|(1+\nu)^{2d(i+1)-d}\\ &\leq (\delta\nu d^2)|S_{\geq i}|(1+\nu)^{2id-d}\end{aligned}$$

where δ and ε are chosen such that $\delta > 2\varepsilon\nu d(1+\nu)^{2d}$. (Note that for c sufficiently large, $(1+\nu)^{2d}$ can be set to an arbitrarily small constant.)

For the induction step we consider two cases. If i is good, then $|S_i| \leq \alpha|S_{>i}|/(2d)$ and $m_i \geq \alpha|S_{>i}|/2$. Therefore, we have

$$\begin{aligned}\Gamma_i &\leq \Gamma_{i+1} + (2\varepsilon\nu^2 d^3)|S_{\geq i}|(1+\nu)^{2id+d-1} - m_i\nu d(1+\nu)^{2id-d}/2\\ &\leq \Gamma_{i+1} + (2\varepsilon\nu^2 d^3)|S_{\geq i}|(1+\nu)^{2id+d-1} - c\nu^2 d^3|S_{>i}|(1+\nu)^{2id-d}/4\\ &\leq \Gamma_{i+1} - (\nu^2 d^3)|S_{\geq i}|(1+\nu)^{2id-d}(f(c,\alpha,d) - 2\varepsilon(1+\nu)^{2d})\\ &\leq (\delta\nu d^2)|S_{>i}|(1+\nu)^{2(i+1)d-d} - (\nu^2 d^3)|S_{\geq i}|(1+\nu)^{2id-d}(f(c,\alpha,d) - 4\varepsilon)\\ &\leq (\delta\nu d^2)|S_{\geq i}|(1+\nu)^{2id-d}((1+\nu)^{2d} - \nu d(f(c,\alpha,d) - 4\varepsilon)/\delta),\end{aligned}$$

where $f(c,\alpha,d) = c/(4(1+\alpha/(2d)))$. (The third inequality follows from the fact that $|S_{>i}| \geq |S_{\geq i}|/(1+\alpha/(2d))$. The fourth inequality follows from the induction hypothesis and the inequality $(1+\nu)^{2d} \leq 2$ for c sufficiently large.)

The second case is when i is bad. Thus $|S_i| \geq \alpha|S_{>i}|/(2d)$. We have

$$\begin{aligned}\Gamma_i &\leq \Gamma_{i+1} + (2\varepsilon\nu^2 d^3)|S_{\geq i}|(1+\nu)^{2id+d-1}\\ &\leq (\delta\nu d^2)|S_{>i}|(1+\nu)^{2(i+1)d-d} + 2\varepsilon\nu^2 d^3|S_{\geq i}|(1+\nu)^{2id+d-1}\\ &\leq (\delta\nu d^2)|S_{\geq i}|(1+\nu)^{2id-d}((1+\nu)^{2d}/(1+\alpha/(2d)) + 2\varepsilon\nu d(1+\nu)^{2d}/\delta).\end{aligned}$$

We now set c, δ, and ε such that $c > 4$, $c/6 - 4\varepsilon \geq 4\delta$, and $c/4 - 2\varepsilon/\delta \geq 4$. (One set of choices is $c = 25$, $\delta = 1$, and $\varepsilon = 1/24$.) Since $\alpha \leq d$, we have $f(c, \alpha, d) \geq c/6$. Since $c > 4$, we have $2\nu d < 1/2$, and hence $(1+\nu)^{2d} \leq 1 + \sum_{i>0}(2\nu d)^i \leq 1 + 2\nu d/(1-2\nu d) \leq 1 + 4\nu d$. Thus,

$$((1+\nu)^{2d} - \nu d(f(c,\alpha,d) - 4\varepsilon)/\delta) \leq 1 + 4\nu d - 4\nu d \leq 1.$$

Since $\alpha/(2d) \leq 1/2$, we have $1/(1+\alpha/(2d)) \leq 1 - \alpha/(4d)$, and hence,

$$\begin{aligned}
(1+\nu)^{2d}/(1+\alpha/(2d)) + \varepsilon\nu d(1+\nu)^{2d}/\delta &\leq (1+\nu)^{2d}(1/(1+\alpha/(2d)) + 2\varepsilon\nu d/\delta) \\
&\leq (1+\nu)^{2d}(1 - \alpha/4d + 2\varepsilon\nu d/\delta) \\
&= (1+\nu)^{2d}(1 - c\nu d/4 + 2\varepsilon\nu d/\delta) \\
&\leq (1+4\nu d)(1-4\nu d) < 1.
\end{aligned}$$

Thus, in both the cases, $\Gamma_i \leq (\delta\nu d^2)|S_{\geq i}|(1+\nu)^{2id-d}$. This completes the inductive step. ■

Corollary 3.5.1 *If j is good on step t then we have $\Psi \geq \varepsilon\nu^2 d^2 \Phi_{t-1}$.*

Proof: Applying Lemma 3.5 with $i = j+1$, it follows that $\Gamma_{j+1} \leq (\delta\nu d^2)|S_{\geq j+1}|(1+\nu)^{2(j+1)d-d}$. If j is good then $|S_j| \leq \alpha|S_{>j}|/(2d)$ and $m_j \geq \alpha|S_{>j}|/2$. Therefore,

$$\begin{aligned}
\Gamma = \Gamma_j &\leq \Gamma_{j+1} + \varepsilon\nu^2 d^3|S_j|(1+\nu)^{2jd+d-1} - \alpha|S_{>j}|(1+\nu)^{2jd+1}/2 \\
&\leq (\delta\nu d^2)|S_{>j}|(1+\nu)^{2(j+1)d-d} + (\alpha\varepsilon\nu^2 d^2)|S_{>j}|(1+\nu)^{2jd+d-1}/2 - \alpha|S_{>j}|(1+\nu)^{2jd+1}/2 \\
&\leq (\nu d^2)|S_{>j}|(1+\nu)^{2(j+1)d-d}(\delta + \varepsilon\alpha^2/(2cd^2) - c/4) \\
&\leq 0,
\end{aligned}$$

for c, δ, and ε chosen above. (In the first inequality, the term $\varepsilon\nu^2 d^3|S_j|(1+\nu)^{2jd+d-1}$ is an upper bound on the contribution to Γ_j by tokens in S_j. The third inequality follows from the fact that for $c > 4$, we have $(1+\nu)^d \leq 2$.)

By the definitions of Γ and Ψ, we have $\Phi_t \leq \Phi_{t-1} - \Psi$ and $\Gamma = \varepsilon\nu^2 d^2 \Phi_{t-1} - \Psi$. If j is good during step t, then $\Gamma \leq 0$, and the desired claim follows. ■

By Corollary 3.5.1, if j is good during step t then we have

$$\Phi_t \leq \Phi_{t-1}(1 - \varepsilon\nu^2 d^2).$$

After $T = \lceil 4\lceil \ln \Phi_0 \rceil/(\varepsilon\nu^2 d^2) \rceil$ steps, we have $\Phi_T \leq \Phi_0(1-\varepsilon\nu^2 d^2)^{T/4} < 1$. Since $\Phi_0 \leq n(1+\nu)^{\Delta+1}/\nu$, $\ln \Phi_0 = O(\Delta\nu + \log n)$. Substituting for ν, we obtain that within $O(\Delta/\alpha + d^2 \ln n/\alpha^2)$ steps, $|S_{>2\log_{1+\alpha/(2d)} n}| \leq |S_{>j}| = 0$.

We use an averaging argument to show that after T steps, $|S_{<-2\log_{1+\alpha/(2d)} n}| \leq n/2$. By redefining the potential function and repeating the above analysis in the other direction, we obtain that in another T steps $|S_{<-4\log_{1+\alpha/(2d)} n}| = 0$. This completes the proof of Theorem 3.2.

3.3 Results in terms of node expansion

The proofs of Theorems 3.1 and 3.2 can be easily modified to analyze the algorithm in terms of the node expansion μ of the graph instead of the edge expansion α. (The primary modifications that need to be done are to alter the definition of a good index, and to set ν appropriately. We set $\nu = \mu/c$ (resp., $\nu = \mu/(cd)$) for the single-port model (resp., multi-port model) and call index i good if $|S_i| \leq \mu|S_{>i}|/2$.)

Theorem 3.3 *For an arbitrary network G with n nodes, maximum degree d, node expansion μ, and initial imbalance Δ, the single-port algorithm load balances within $O((\log n)/\mu)$ tokens in $O(d\Delta/\mu)$ steps with high probability.*

Corollary 3.3.1 *If $\Delta \geq (d \log n)/\mu$, the single-port algorithm balances to within $O(\log n/\mu)$ tokens in $O((d\Delta)/\alpha)$ steps with high probability. If $\Delta < (d \log n)/\mu$, the single-port algorithm balances to within $O(\log n/\mu)$ tokens in $O((d\Delta)/\mu)$ steps with high probability.*

Theorem 3.4 *For an arbitrary network G with n nodes, maximum degree d, node expansion μ, and initial imbalance Δ, the multi-port algorithm load balances to within $O((d \log n)/\mu)$ tokens in $O(\Delta/\mu)$ steps.*

Corollary 3.4.1 *If $\Delta \geq (d^2 \log n)/\mu$, the multi-port algorithm balances to within $O((d \log n)/\mu)$ tokens in $O(\Delta/\alpha)$ steps. If $\Delta < (d^2 \log n)/\mu$, the multi-port algorithm balances to within $O((d \log n)/\mu)$ tokens in $O(\Delta/\mu)$ steps.*

4 Local load balancing can be expensive

In this section, we show that locally load balancing to within $2d$ tokens using the multi-port algorithm of [2] described in Section 1.1 can take $\Omega(n^{1/2})$ more time than globally load balancing to within $O((d \log n)/\mu)$ tokens. We extend this bound for the single-port algorithm presented in [14], i.e., the expected number of additional steps this algorithm may take to perform local (to within one token) rather than global balancing is $\Omega(dn^{1/2})$. Furthermore, we show that in the single-port case, we may be one step away from being locally balanced to within one token but have an expected running time of $\Omega(\mu n^{1/2})$ for reaching a locally balanced state. Finally, we prove upper bounds on the time each algorithm takes to reach a locally balanced state.

We will now construct a graph $G = (V, E)$ on n nodes and show that this graph has expansion $\Omega(\mu)$. Let $0 < \mu \leq 1$ (we will actually need $\mu \leq 1/3$, as seen below). Let $V = (\cup_{i=0}^{k} L_i) \cup (\cup_{i=0}^{k-1} R_i)$, where L_i and R_i are sets of $(1+\mu)^i$ nodes each. For convenience, let L_{-1} (resp., R_{-1}) denote R_0 (resp., L_0) and R_k denote L_k. Note that each L_i and each R_i has size $(1+\mu)^i = \mu(\sum_{j=0}^{i-1} |L_j|) + 1 = \mu(\sum_{j=0}^{i-1} |R_j|) + 1$. Thus $k = O(\log n/\log(1+\mu)) = O(\log n/\mu)$. Here we assume w.l.o.g. that $k = \log n/\log(1+\mu)$ and that k is even. Let $0 < \mu' \leq 1/3$ and $\mu \leq \mu'$.

For each i, $0 \leq i \leq k$, let every subset S of L_i of size at most $3|L_i|/4$ have at least $\mu'|S|$ neighbors in $L_i \setminus S$, and let L_i have maximum degree $d/2$, for some suitable value of d, which depends on μ'. Let each node in L_i have exactly $d(1+\mu)/(2(2+\mu))$ neighbors in L_{i+1} and each node in L_{i+1} have exactly $d/(2(2+\mu))$ neighbors in L_i, $0 < i \leq k-1$ (note that w.l.o.g. we will ignore integrality constraints, since we can always find many values of d and μ such that integrality holds). Using a counting argument on the number of edges between levels that are adjacent to each node, we see that every $S \subseteq L_i$ has at least $(1+\mu)|S|$ neighbors in L_{i+1}. Now we consider how L_{i+1} "expands" into L_i. We can use a similar approach as in [23, 37] to show that we can choose the edges between L_i and L_{i+1}, respecting the degree constraints above, such that any $S \subseteq L_{i+1}$ such that $|S| \leq |L_{i+1}|/(1+\mu)^2$ has at least $(1+\mu)|S|$ neighbors in L_i. Let the only node in L_0 be adjacent to every node in L_1. We obtain a similar structure for the R_i's by replacing L_j by R_j in this paragraph. Note that the maximum degree of G is $\leq d$.

Lemma 4.1 *G has node expansion $\Omega(\mu\mu')$.*

Proof: Let $S \subset V$, $|S| \leq n/2$. Let $S_i = S \cap L_i$ and $S'_i = S \cap R_i$, $0 \leq i \leq k$. Let $N_Y(X) = \{y \in Y | (x,y) \in E, x \in X\}$, for any $X, Y \subseteq V$; if the set Y is not specified, assume $Y = V$. Let $0 \leq i \leq k$:

1. If $|S_i| \leq 3|L_i|/4$, then S_i has at least $\mu'|S_i|$ neighbors inside $L_i \setminus S_i$.
2. If $i < k$ and $|S_i| > 2|L_i|/3$ and $|S_{i+1}| \leq 2|L_{i+1}|/3$, then:
   (a) if $2|L_i|/3 < |S_i| \leq 3|L_i|/4$ then $|N_{L_i}(S_i)| - |S_i| \geq \mu' 2|L_i|/3 = (\mu'/3)(2|L_i|) > (\mu'/3)(\mu \sum_{j=0}^{i-1} |L_j| + |L_i|) \geq (\mu\mu'/3)(\sum_{j=0}^{i} |L_j|)$;

(b) if $|S_i| > 3|L_i|/4$ then $|N_{L_{i+1}}(S_i)| - |S_{i+1}| > (1+\mu)3|L_i|/4 - 2|L_{i+1}|/3 = 3|L_{i+1}|/4 - 2|L_{i+1}|/3 = |L_{i+1}|/12 > (\mu/12)(\sum_{j=0}^{i}|L_j|)$

Thus $|N(S_i)| - |S| \geq (\mu\mu'/12)(\sum_{j=0}^{i}|S_j|)$.

Similar results hold if we replace S_j by S'_j and L_j by R_j in the items above.

Now we consider every S_i, (resp., S'_i) such that $|S_i| > 2|L_i|/3$ (resp., $|S'_i| > 2|R_i|/3$) but there is no $z > i$ such that $|S_z| \leq 2|L_z|/3$ (resp., $|S'_z| \leq 2|R_z|/3$). Let $i_\ell = max_i\{i > 1 \mid |S_i| > 2|L_i|/3$ and $|S_{i-1}| \leq 2|L_{i-1}|/3\}$ and $i_r = max_j\{j > 1 \mid |S'_j| > 2|R_j|/3$ and $|S'_{j-1}| \leq 2|R_{j-1}|/3\}$. Let $i_\ell = 0$ (resp., $i_r = 0$), in case no such i (resp., j) exists. If i_ℓ and i_r are not both equal to 0, then we must have either $\sum_{i=0}^{i_\ell-1}|L_i| \geq n/8$ or $\sum_{i=0}^{i_r-1}|R_i| \geq n/8$ (since $|S| \leq n/2$). Assume w.l.o.g. that the former is true. This implies that $|L_{i_\ell}| > \mu n/8$. Then:

a. if $|S_{i_\ell}| \geq |L_{i_\ell}|/(1+\mu)^2$ then $N_{L_{i_\ell-1}}(S_{i_\ell}) = L_{i_\ell-1}$ and $|N_{L_{i_\ell-1}}(S_{i_\ell})| - |S_{i_\ell-1}| \geq |L_{i_\ell-1}|/3 = |L_{i_\ell}|/(3(1+\mu)) > \mu(\sum_{j=0}^{i_\ell-1}|L_j|)/(3(1+\mu)) \geq \mu n/48 = (\mu/24)(n/2)$.

b. if $|L_{i_\ell}|/(1+\mu)^2 > |S_{i_\ell}| > 3|L_{i_\ell}|/4$ then $|N_{L_{i_\ell-1}}(S_{i_\ell})| - |S_{i_\ell-1}| \geq (1+\mu)|S_{i_\ell}| - 2|L_{i_\ell-1}|/3 > (1+\mu)3|L_{i_\ell}|/4 - 2|L_{i_\ell}|/(3(1+\mu)) > (1+\mu)3|L_{i_\ell}|/4 - 2|L_{i_\ell}|/3 > |L_{i_\ell}|/12 > (\mu n/8)/12 = (\mu/48)(n/2)$;

c. if $|S_{i_\ell}| \leq 3|L_{i_\ell}|/4$ then $|N_{L_{i_\ell}}(S_{i_\ell})| - |S_{i_\ell}| \geq \mu'|S_{i_\ell}| > \mu' 2|L_{i_\ell}|/3 \geq (2\mu'/3)(\mu n/8) = (\mu\mu'/6)(n/2)$.

Hence, since we count each neighbor of S at most three times and we account for each S_i and each S'_i at least once in items (1) and (2), and in the preceding paragraph, S has $\Omega(\mu\mu'|S|)$ neighbors outside S in G. ∎

Corollary 4.1.1 *If $\mu' = 1/3$ then G has node expansion $\Omega(\mu)$, $0 < \mu \leq 1/3$.*

Let $L_0 = \{u\}$ and $R_0 = \{v\}$. Let $G' = (V, E')$ be the graph obtained by adding an edge e connecting v to u to G (see Figures 1, 2). Note that the node expansion of G cannot be reduced by the addition of new edges and so the node expansion of G' is also $\Omega(\mu\mu')$ ($= \Omega(\mu)$, if we fix μ'). We state all the results in this section in terms of the node expansion of the network, rather than in terms of its edge expansion. This is done for the sake of making our arguments more intuitive and clear.

We present the cases below according to the initial distribution of tokens over the nodes of G'. We use $w(x)$ to denote the number of tokens on node x. We will use the fact that at any time step every node in L_i (resp. R_i), $0 \leq i \leq k$, has exactly the same number of tokens, if they had the same number of tokens initially. We prove this using induction. W.l.o.g. we will prove it for sets L_i only. The same proof holds if we replace L_j by R_j below. Suppose every node in L_i had the same number of tokens at time step $t-1$. A node $x \in L_i$ only sends a token to one of its neighbors in L_{i+1} when it has at least $2d+1$ more tokens than its neighbor has. Thus if at time t, x sends a token to some $y \in L_{i+1}$ then it sends a token to all its neighbors in that set, since all of them had the same number of tokens at time $t-1$ and the number of neighbors of x in L_{i+1} is at most d. Hence at time t, every edge between L_i and L_{i+1} is traversed by a token. Since every node in L_{i+1} sees exactly the same number of nodes in L_i, they will have all received the same number of tokens from L_i. We can use a similar argument for tokens that move from L_i to L_{i-1}. No token moves from a node in L_i to another node inside L_i, since all nodes in L_i had the same number of tokens at time $t-1$. When $i = k$, only consider tokens moving from L_k and R_k to L_{k-1} and R_{k-1}, resp.. From the remark above, we see that tokens never move inside each L_i or R_i, and so we ignore the edges inside each of these sets.

We group the sets R_i's and L_i's into $\mathcal{L}$ and $\mathcal{R}$, groups of $k/2$ consecutive sets, and $\mathcal{M}$, a group of $k+1$ consecutive sets (note that we have $2k+1$ distinct sets). Let $\mathcal{L} = \{L_0, L_1, \ldots, L_{k/2-1}\}$, $\mathcal{R} = \{R_0, R_1, \ldots, R_{k/2-1}\}$ and $\mathcal{M} = \{L_{k/2}, L_{k/2+1}, \ldots, L_{k-1}, R_k(= L_k), R_{k-1}, \ldots, R_{k/2}\}$. Our choice for $\mathcal{L}, \mathcal{M}$ and $\mathcal{R}$ is such that the number of sets in $\mathcal{L}$ is roughly half the number of sets in $\mathcal{M}$.

4.1 The local algorithm may take $\Omega(n^{1/2})$ steps to locally balance G'

Suppose for every node z in $\mathcal{R}$, $w(z) = m + 1$ initially, where m is an integer such that $m \geq 2kd$. For all $z \in R_i$, $R_i \in \mathcal{M}$, let $w(z) = m - 2(i - k/2)d$; for all $z \in L_i$, $L_i \in \mathcal{M}$, let $w(z) = m - 2(3k/2 - i)d$. For all $z \in L_i$, $L_i \in \mathcal{L}$, let $w(z) = m - 2(i + k/2 + 1)d$. Then w is globally balanced to within $O((d \log n)/\mu)$ tokens, but it is not locally balanced to within $2d$ tokens since $w(v) - w(u) > kd$ $(\geq 2d)$. See Figure 1.

There are at least $(1+\mu)^{k/2-1} = (1+\mu)^{\frac{\log n}{2\log(1+\mu)}-1} = \sqrt{n}/(1+\mu) \geq \sqrt{n}/2$ nodes in $\mathcal{R}$, as there are in $\mathcal{L}$. We claim that in order for G' to be locally balanced to within $2d$, we need to move from $\mathcal{R}$ to $\mathcal{L}$ at least $\sqrt{n}/2$ tokens across edge e. Since at most one token at a time can traverse e, it will take time $\Omega(n^{1/2})$ to locally balance G' to within $2d$ tokens. Our proof proceeds as follows:

i. By observing how the tokens flow in G' - since every node in R_j (resp., L_j) is identical with respect both to the number of tokens it has and to the number of neighbors it sees in R_{j-1} and R_{j+1} (resp., L_{i-1} and L_{i+1}) - we can see that no token ever moves from left to right (i.e. from L_i to L_{i-1} or from R_i to R_{i+1}) in G'. In particular, no token ever moves from $\mathcal{R}$ to $\mathcal{M}$ or from $\mathcal{M}$ to $\mathcal{L}$.

ii. Now we show that we can only have $w(u) > m - (k+2)d$ and $w(x) - w(y) \leq 2d$, $\forall x \in L_i$, $\forall y \in L_{i+1}$, $\forall L_i \in \mathcal{L}$ after $\sqrt{n}/2$ steps. Suppose we reach such a configuration for w at some time t. Then every node in $\mathcal{L}$ has at least one more token than it had initially. That is, we have at least $|\mathcal{L}| \geq \sqrt{n}/2$ "extra" tokens in $\mathcal{L}$ at time t, all of which must have reached $\mathcal{L}$ by traversing e from v to u, since no token moves from $\mathcal{M}$ to $\mathcal{L}$. And since only one token at a time can traverse e, we have $t \geq \sqrt{n}/2$.

iii. We also show that we can only have: $w(v) \leq m - kd$ and $w(y) - w(x) \leq 2d$, $\forall x \in R_i, \forall y \in R_{i+1}, \forall R_i \in \mathcal{R}$ after $\sqrt{n}/2$ steps. A counting argument (similar to the one above) on the number of tokens in $\mathcal{R}$ and the fact that no token is ever sent from $\mathcal{R}$ to $\mathcal{M}$, is sufficient to show this.

It follows from (ii) and (iii) that either v sends a token to u, or $\mathcal{R}$ or $\mathcal{L}$ is not $2d$-locally balanced in any of the first $\sqrt{n}/2$ steps, and so G' is not locally balanced before these steps. Hence the algorithm will take additional time $\Omega(\sqrt{n}/2)$ to locally balance G'.

Note that the arguments above can be easily modified in order to hold for the single-port model (divide the number of tokens above height m by $2d$ on each node not in $\mathcal{R} \cup R_{k/2}$, consider differences of at most one token between "adjacent" R_i's or L_i's, and consider randomization), since the lower bound on the number of steps required to reach a locally balanced state is given only in terms of how many tokens need to traverse the edge e. Any edge is selected with probability $O(\frac{1}{d})$ at each iteration of the single-port algorithm. Thus the expected time for e being selected at some time step is $\Omega(d)$. Hence we have that it will take $\Omega(dn^{1/2})$ expected time for G', if G' is globally balanced to within $O((\log n)/\mu)$ tokens, to reach a locally balanced state to within 1 token in the single-port model.

4.2 The local algorithm may take one step or $\Omega(\mu n^{1/2})$ expected time to reach local balance

Here we consider the single-port model. Suppose we have the following distribution of tokens: for all $z \in R_{k/2-1}$, let $w(z) = m + 1$ (where m is an integer such that $m \geq k$); for all $z \in R_i$, $i \leq k/2 - 2$ (note that $R_i \in \mathcal{R}$), let $w(z) = m - (k/2 - i - 1)$; for all $z \in R_{k/2}$, let $w(z) = m - 1$; for all $z \in R_i$, $i \geq k/2 + 1$ (note that $R_i \in \mathcal{M}$), let $w(z) = m - (i - k/2)$; for all $z \in L_i$, $L_i \in \mathcal{M}$, let $w(z) = m - (3k/2 - i)$. For all $z \in L_i$, $L_i \in \mathcal{L}$, let $w(z) = m - (k/2 + i)$. Thus w is globally balanced to within $O((\log n)/\mu)$ tokens but it is not locally balanced to within 1 token, since $w(x) - w(y) > 1$, for any $x \in R_{k/2-1}$ and $y \in R_{k/2} \cup R_{k/2-2}$. See Figure 2.

The intuition for this case is that if all tokens move in the "right direction" initially, we reach a locally balanced state in a single time step. Otherwise, if a large number of tokens move in the "wrong direction" in the first step, it will take a long time to reach such a state. If every node in $R_{k/2-1}$ is matched with some node in $R_{k/2}$ (we can assume that every node in $R_{k/2-1}$ has a distinct neighbor in $R_{k/2}$), G' reaches local balance in a single time step. On the other hand, if we move tokens across a matching between the nodes in $R_{k/2-1}$ and $R_{k/2-2}$ then these tokens will clearly continue moving "down" (non-decreasing indexes of R_i), and never move "up". The expected size of such a matching will be $\Omega(|R_{k/2-1}|/d)$ (each node in $R_{k/2-1}$ has exactly $d/(2(2+\mu)) \geq d/5$ neighbors in $R_{k/2-2}$). Using a similar analysis (taking care of randomization) as in the previous case, we see that no token ever moves from $\mathcal{M}$ to $\mathcal{L}$ and that $w(v) > m - k/2 + 1$ and $w(u) = m - k/2$, or $\mathcal{L} \cup \mathcal{R}$ is not locally balanced, for each of the $\Omega(|R_{k/2-1}|/d)$ initial steps. Thus, once any of these tokens (that moved from $R_{k/2-1}$ to $R_{k/2-2}$ in the first step) reaches v, it will eventually traverse e onto u.

Hence, eventually all tokens that moved from $R_{k/2-1}$ to $R_{k/2-2}$ in the initial step will reach u. Since $|R_{k/2-1}| \geq \frac{\mu n^{1/2}}{(1+\mu)^2}$ and $1 \leq (1+\mu)^2 \leq 2$, and since the expected time for edge e to be in the selected matching is $\Omega(d)$, the expected time for G' to reach local balance to within one token is $\Omega(\mu n^{1/2})$.

4.3 Convergence to a locally balanced state

We now prove that if the graph G is globally balanced to within Δ tokens, in $O(n\Delta^2/d)$ subsequent steps the multi-port algorithm locally balances G to within $2d$ tokens. Define the potential Φ of the network as $\sum_{v \in V}(w(v) - \rho)^2$. If the network is globally balanced to within Δ tokens, then $\Phi = O(n\Delta^2)$. At any step, if there exists an edge (u, v) such that $|w(u) - w(v)| \geq 2d+1$, then a token is transmitted along (u, v) resulting in a potential drop of at least d. Thus, within $O(n\Delta^2/d)$ steps the network is globally balanced. Similarly, we show that if the graph G is globally balanced to within Δ tokens, then the single-port algorithm locally balances to within 1 token in $O(nd\Delta^2)$ subsequent steps with high probability.

5 Extension to dynamic and asynchronous networks

In this section, we extend our results for the multi-port model of Section 3.2 to dynamic and asynchronous networks. We first prove that a variant of the local multi-port algorithm is optimal on dynamic synchronous networks in the same sense as for static synchronous networks. We then use a result of [2] that relates the dynamic synchronous and asynchronous models, to extend our results to asynchronous networks.

In the dynamic synchronous model, the edges of the network may fail or succeed dynamically. An edge $e \in E$ is *live* during step t if e can transmit a message in each direction during step t. We assume that at each step each node knows which of its adjacent edges are live. The local load balancing algorithm for static synchronous networks can be modified to work on dynamic synchronous networks. The algorithm presented here is essentially the same as in [2]. Every node u maintains an estimate $e^u(v)$ of the number of tokens at v for every neighbor v of u. (The value of $e^u(v)$ at the start of the algorithm is arbitrary.) In every step of the algorithm, which we call **DS**, each node u does the following:

(1) For each live neighbor v of u, if $w(u) - e^u(v) > 12d$, u sends a message consisting of $w(u)$ and a token, otherwise u sends a message consisting only of $w(u)$.

(2) For each message received from a live neighbor v, $e^u(v)$ is updated according to the message and if the message contains a token, $w(u)$ is increased by one.

For every integer i, we denote the set of nodes having between $\rho - 12d + 24id$ and $\rho + 12d - 1 + 24id$ tokens as S_i. Consider T steps of **DS**. We assume without loss of generality that $|S_{>0}| \leq n/2$ at the start of at least $T/2$ steps. There exists an index j in $[1, \lceil 2\log_{1+\alpha/(2d)} n \rceil]$ that is good in at least half the steps at the start of which $|S_{>0}| \leq n/2$. If index j is good at the start of step t, we call t a *good* step. For any token p, let $h_t(p)$ denote the height of p after step t, $t > 0$. For convenience, we denote the height of p at the start of **DS** by $h_0(p)$. Similarly, for $t \geq 0$, we define $h_t(u)$ for every node u and $e^u_t(v)$ for every edge (u, v).

With every token at height h, we associate a potential of $\phi(h)$, where $\phi : N \to R$ is defined as follows:

$$\phi(x) = \begin{cases} 0 & \text{if } x \leq 24jd - 11d, \\ (1+\nu)^x & \text{otherwise,} \end{cases}$$

where $\nu = \alpha/(cd^2)$, and $c > 0$ is a constant to be specified later. Let Φ_t denote the total potential of the network after step t. Let Ψ_t denote the potential drop during step t.

We analyze **DS** by means of an amortized analysis over the steps of the algorithm. Let E_t be the set $\{(u,v) : (u,v) \text{ is live, } u \in S_{>j} \text{ and } h_{t-1}(u) - h_{t-1}(v) \geq 24d\}$. For every step t, we assign an amortized potential drop of

$$\hat{\Psi}_t = \frac{1}{50} \sum_{\substack{(u,v)\in E_t \\ h_{t-1}(u) > h_{t-1}(v)}} (\phi(h_{t-1}(u) - d) - \phi(h_{t-1}(v) + d)).$$

The definition of $\hat{\Psi}_t$ is analogous to the amount of potential drop that we use in step t in the argument of Section 3.2 for the static case. By modifying that argument slightly and choosing appropriate values for the constants c and ε, we can show the following lemma.

Lemma 5.1 *If the live edges of G have an edge expansion of α during every step of* **DS**, *then for every good step t we have $\hat{\Psi}_t \geq \varepsilon\nu^2 d^2 \Phi_{t-1}$, where ε is a constant chosen appropriately.*

Proof Sketch: Let M_i denote the set of live edges between nodes in $S_{<i}$ and nodes in $S_{>i}$. Let $m_i = |M_i|$. For any node u, let $g(u)$ represent the group to which u belongs prior to step t. We now place a lower bound on $\hat{\Psi}_t$ which is analogous to that on Ψ in Lemma 3.4 of Section 3.2. By the definition of $\hat{\Psi}_t$, we have

$$\begin{aligned}
\hat{\Psi}_t &= \frac{1}{50} \sum_{\substack{(u,v)\in E_t \\ (u,v)\notin M_j \\ h_{t-1}(u) > h_{t-1}(v)}} (\phi(h_{t-1}(u) - d) - \phi(h_{t-1}(v) + d)) + \frac{1}{50} \sum_{\substack{(u,v)\in E_t \\ (u,v)\in M_j \\ h_{t-1}(u) > h_{t-1}(v)}} (\phi(h_{t-1}(u) - d) - \phi(h_{t-1}(v) + d)) \\
&\geq \frac{1}{50}\left(\sum_{\substack{(u,v)\in E_t \\ (u,v)\notin M_j \\ h_{t-1}(u) > h_{t-1}(v)}} \sum_{g(v)<i<g(u)} (\phi(24(i+1)d - 13d) - \phi(24(i-1)d + 13d)) \right) \\
&\quad + \frac{1}{50} \sum_{\substack{(u,v)\in E_t \\ (u,v)\in M_j \\ h_{t-1}(u) > h_{t-1}(v)}} \sum_{j \leq i < g(u)} (\phi(24(i+1)d - 13d) - \phi(24(i-1)d + 13d)) \\
&\geq \frac{22}{50}\left(\sum_{\substack{i>j \\ (u,v)\in M_i \\ h_{t-1}(u) > h_{t-1}(v)}} (\phi(24(i+1)d - 13d) - \phi(24(i-1)d + 13d)) \right) \\
&\quad + \frac{1}{50} \sum_{\substack{(u,v)\in M_j \\ h_{t-1}(u) > h_{t-1}(v)}} \phi(24(j+1)d - 13d) \\
&\geq \frac{22}{50}\left(\sum_{\substack{i>j \\ (u,v)\in M_i \\ h_{t-1}(u) > h_{t-1}(v)}} \nu d(1+\nu)^{24id-11d} \right) + \frac{1}{50} \sum_{\substack{(u,v)\in M_j \\ h_{t-1}(u) > h_{t-1}(v)}} (1+\nu)^{24jd+11d}
\end{aligned}$$

$$\geq \frac{22}{50}\left(\sum_{i>j} m_i \nu d(1+\nu)^{24id-11d}\right) + \frac{1}{50} m_j (1+\nu)^{24jd+11d}.$$

(The second step can be verified by expanding the two inner summations indexed by i. The third step is obtained from the second step by combining the two summations and redistributing the terms according to the value of index i. In the fourth step we use the inequality $(1+x)^{22d} \geq 1 + 22dx$ for $x \geq 0$.)

We can then establish claims similar to Lemma 3.5 and Corollary 3.5.1 of Section 3.2 by just modifying the constants in the proofs. Thus we have $\hat{\Psi}_t \geq \varepsilon \nu^2 d^2 \Phi_{t-1}$ for constant ε chosen appropriately. ■

The following crucial lemma relates the amortized potential drops to the actual potential drops.

Lemma 5.2 *For any initial load distribution and any step $t' > 0$, we have*

$$\sum_{t \leq t'} \hat{\Psi}_t \leq \left(\sum_{t \leq t'} \Psi_t\right) - 2\Phi_0 - n^2 \phi(24jd).$$

The main result follows from Lemmas 5.1 and 5.2. We first show that within $O(1/(\varepsilon\nu^2 d^2))$ steps, there is a step when the actual potential of the network either decreases by a factor of 2 or falls below a threshold value.

Lemma 5.3 *Let t be any integer such that at least $7/(\varepsilon\nu^2 d^2)$ of the first t steps are good. There exists $t' \leq t$ such that $\Phi_{t'} \leq \max\{\Phi_0/2, n^2\phi(24jd)\}$.*

Proof: If $\Phi_0 \leq n^2\phi(24jd)$, then the claim is proved for $t = 0$. For the remainder of the proof, we assume that $\Phi_0 \geq n^2\phi(24jd)$. If $\Phi_{t'} \leq \Phi_0/2$ for any $t' < t$, the claim is proved. Otherwise, for all $t' < t$, we have $\Phi_{t'} > \Phi_0/2$. In this case, we obtain

$$\begin{aligned}
\Phi_t &= \Phi_0 - \sum_{t' \leq t} \Psi_{t'} \\
&\leq 3\Phi_0 + n^2\phi(24jd) - \sum_{t' \leq t} \hat{\Psi}_{t'} \\
&\leq 4\Phi_0 - \sum_{\substack{t' \leq t \\ t' \text{good}}} (\varepsilon\nu^2 d^2)\Phi_{t'} \\
&\leq \Phi_0/2.
\end{aligned}$$

(In the second step, we invoke Lemma 5.2. In the third step, we invoke Lemma 5.1 and use the inequalities $\Phi_0 \geq n^2\phi(24jd)$, and $\hat{\Psi}_{t'} \geq 0$ for every t'. In the fourth step, we use the fact that at least $7/(\varepsilon\nu^2 d^2)$ of the t steps are good and the inequality $\Phi_{t'} > \Phi_0/2$ for every $t' \leq t$.) ■

Theorem 5.1 *For an arbitrary network G with n nodes, degree d and initial imbalance Δ, if the live edges at every step t of G have edge expansion α, then the dynamic synchronous multi-port algorithm load balances to within $O(d^2(\log n)/\alpha)$ tokens in $O(\Delta/\alpha)$ steps.*

Proof: By repeatedly invoking Lemma 5.3, we obtain that within $\lceil (7/(\varepsilon\nu^2 d^2)) \rceil \lceil \log \Phi_0 \rceil$ good steps, there exists a step after which the potential of the network is at most $n^2\phi(24jd)$. (Note that the fact that Lemma 5.2 holds for arbitrary initial values of the estimates, the $e^u(v)$'s, is crucial here.) Since at least $T/4$ of the first T steps are good, there exists $t \leq 4\lceil (7/(\varepsilon\nu^2 d^2)) \rceil \lceil \log \Phi_0 \rceil$ such that $\Phi_t \leq n^2\phi(24jd)$. Since $\Phi_0 \leq n(1+\nu)^{(\Delta+1)}/\nu$, we have $\log \Phi_0 \leq (\log n + (\Delta+1)\log(1+\nu) - \log \nu)$. Since $\nu = \alpha/(cd^2)$ and $\log(1+\nu) = O(\nu)$, we have $t = O((\Delta/\alpha) + d^2(\log n)/(\alpha^2))$.

Let h be the maximum height of any node after step t. We thus have

$$\begin{aligned}\phi(h) &\leq \Phi_t\\ &\leq n^2(1+\nu)^{24jd}.\end{aligned}$$

Therefore $h \leq \max\{24jd - 11d, \log_{(1+\nu)}(n^2(1+\nu)^{24jd})\}$. Since $\log(1+\nu) < \nu/2$ for c appropriately large, we have

$$\begin{aligned}h &\leq 24jd + (2\log n)/\log(1+\nu)\\ &\leq 24jd + (4\log n)/\nu\\ &= O((d^2\log n)/\alpha)\end{aligned}$$

(The last step follows from the relations $\nu = \alpha/(cd^2)$ and $j = O((d\log n)/\alpha)$.) Thus, at the end of step t, no node has more than $a = \rho + h$ tokens. We now prove by contradiction that for every step $t > t'$, no node has more than $a + d$ tokens. Let t' be the first step after step t such that there exists some node u with more than $a+d$ tokens. (If no such t' exists, the claim holds trivially.) Of the $d+1$ highest tokens received by u after step t, at least 2 tokens (say p and q) were last sent by the same neighbor v of u. Without loss of generality, we assume that p arrived at u before q. Let t_1 be the step when p was last sent by v to u. Therefore, we have $e^v_{t_1}(u) \geq h_{t_1}(p) - d \geq a - d$. Hence q can be sent to u only when v has height at least $a + 11d$, which contradicts our choice of t'.

We have shown that after $O(\Delta/\alpha + (d^2\log n)/\alpha^2)$ steps, no node ever has more than $\rho + O((d^2\log n)/\alpha)$ tokens. An easy averaging argument shows that there exists $k = O((d\log n)/\alpha)$ such that after every step $t' \geq t$, $|S_{<-k}| \leq n/2$. By defining an appropriate potential function for tokens with heights below the average and repeating the analysis done for $S_{>j}$, we show that in another $O((\Delta/\alpha) + d^2(\log n)/(\alpha^2))$ steps, all nodes have more than $\rho - O(d^2(\log n)/\alpha)$ tokens. ■

We now prove Lemma 5.2. In order to do this, we need to address two issues that arise in the dynamic setting: (i) potential gains, i.e., when a token gains height, and (ii) no actual potential drops across edges that join nodes differing by at least $24d$ tokens. We show that for any of the above events to occur, "many" tokens should have lost height in previous steps. We use a part of this prior potential drop to account for (i) and (ii). We now define a notion of *goodness* of the tokens. Initially, all tokens are unmarked. After any step t, for every token p that is moved along an edge, p is marked *good* if $h_t(p) - h_{t-1}(p) \geq 6d$; otherwise p is marked *bad*. The marking of tokens that do not move is unchanged.

Lemma 5.4 *For any two bad tokens p_1 and p_2 present at any node u at the start of any step t, if p_1 and p_2 were last sent to u by the same neighbor v of u, then $|h_t(p_1) - h_t(p_2)| > 4d$.*

Proof: Let t_1 (resp., t_2) be the steps during which p_1 (resp., p_2) are last sent to u. Without loss of generality, we assume $t_1 < t_2$. Thus we have $h_t(p_1) < h_t(p_2)$. By the definition of the algorithm, $e^v_{t_1}(u) \geq \rho + h_{t_1}(p_1) - d$. Since p_1 remains at u during the interval $[t_1, t_2)$, we find that $e^v_{t'}(u) \geq \rho + h_{t'}(p_1) - d$ for every step t' in $[t_1, t_2)$. Therefore, $h_{t_2-1}(p_2) \geq h_{t_2-1}(v) - d \geq e^v_{t_2-1}(u) + 11d \geq h_{t_2-1}(p_1) + 10d$. Since p_2 is bad, we also have $h_{t_2}(p_2) > h_{t_2-1}(p_2) - 6d \geq h_{t_2-1}(p_1) + 4d$. Since $h_t(p_2) = h_{t_2}(p_2)$ and $h_t(p_1) = h_{t_2-1}(p_1)$, the lemma follows. ■

Corollary 5.4.1 *At any time, for any node u and integer $i > 0$, there are at most d bad tokens with heights in $[i, i+4d]$.* ■

Proof of Lemma 5.2: Consider an arbitrary step t of the algorithm. For every token p transferred from u to v in step t, we assign some credit to every edge adjacent to u or v. Specifically, if p is marked good after step t we assign a credit of $9(\phi(h_t(p) + 6d) - \phi(h_t(p)))/(20d)$ units to every edge adjacent to u

or v. If p is marked bad we assign a credit of $(\phi(h_t(p)+d)-\phi(h_t(p)))/(20d)$ units to every edge adjacent to u or v. For a token transferred from u to v, the credit assigned to edges adjacent to u (resp., v) is called *outgoing credit* (resp., *incoming credit*). Also, for each pair (u,v), we assign an *initial credit* of $2\max\{\phi(24jd),(\phi(h_0(u)-d)+\phi(h_0(v)-d))\}$ units at the start of the analysis. The total initial credit I is bounded as follows:

$$
\begin{aligned}
I &\le 2\binom{n}{2}\phi(24jd)+\sum_{(u,v)\in E}2(\phi(h_0(u)-d)+\phi(h_0(v)-d))\\
&\le n^2\phi(24jd)+\sum_{u\in V}\sum_{0\le \ell<d}2\phi(h_0(u)-\ell)\\
&\le n^2\phi(24jd)+2\Phi_0.
\end{aligned}
$$

(This bound corresponds to the negative term in Equation (5.2).)

We now show that using the above accounting method, we can account for the amortized potential drop of $(\phi(h_{t-1}(u)-d)-\phi(h_{t-1}(v)+d))/50$ units at step t for every live edge $(u,v)\in E_t$. To accomplish this, for every live edge (u,v) ((u,v) not necessarily in E_t), we consider three cases: (i) a token p sent from u to v is marked good, (ii) a token p sent from u to v is marked bad, (iii) no token is sent from u to v.

We first consider case (i). When a token p is marked good after being sent along (u,v), we pay for the amortized drop D_1 as well as the credit D_2 associated with p from the actual potential drop of p:

$$
\begin{aligned}
D_1+D_2 &= 2d[9(\phi(h_t(p)+6d)-\phi(h_t(p)))]/(20d)+(\phi(h_{t-1}(u)-d)-\phi(h_{t-1}(v)+d))/50\\
&\le 9(\phi(h_{t-1}(p))-\phi(h_t(p)))/10+(\phi(h_{t-1}(p))-\phi(h_t(p)))/50\\
&\le \phi(h_{t-1}(p))-\phi(h_t(p)).
\end{aligned}
$$

(The second step follows from the fact that p is a good token which implies that $h_t(p)\le h_{t-1}(p)-6d$.)

We now consider case (ii). In this case we need to account for: (1) if $h_t(p)>h_{t-1}(p)$, an amount equal to the potential increase of $D_1=\phi(h_t(p))-\phi(h_{t-1}(p))$ units, and (2) a credit of $D_2\le(\phi(h_t(p)+d)-\phi(h_t(p)))/10$ units. We have two subcases, depending on whether t is the first step (u,v) is live (subcase (a)) or not (subcase (b)). In subcase (a), if $h_0(u)\ge h_t(p)-d$, the initial credit C_0 associated with (u,v) is at least $2\max\{\phi(24jd),\phi(h_t(p)-2d)\}$. It follows from Corollary A.1.1 that $3C_0/4\ge\phi(h_t(p))$ and $C_0/4\ge\phi(h_t(p)+d)/10$. Therefore, we have $C_0\ge D_1+D_2$.

We now consider subcase (a) under the assumption that $h_0(v)\le h_t(p)-d$. In order to do the accounting, we use part of the incoming credit associated with the edge (u,v) due to the set X of good tokens of v with heights in the interval $[h_0(v),h_t(p)-d]$. (Note that each token in X is marked and thus, has contributed incoming credit to all edges adjacent to v.) For each token x in X, we use c_x units of credit, which equals $8(\phi(h_t(q)+6d)-\phi(h_t(q)))/(20d)$ units of the incoming credit assigned to (u,v) by x. Let C_1 denote $\sum_{x\in X}c_x$. By invoking Corollary 5.4.1, we obtain:

$$
\begin{aligned}
C_1 &\ge \sum_{1\le i\le\lfloor\frac{h_t(p)-d-h_0(v)}{4d}\rfloor}(\phi(h_t(p)-d-4id+6d)-\phi(h_t(p)-d-4id))\\
&\ge (3\cdot 8)(\phi(h_t(p)+d)-\phi(h_0(v)+4d))/20\\
&= 6(\phi(h_t(p)+d)-\phi(h_0(v)+4d))/5.
\end{aligned}
$$

Since p is marked bad after step t, we have $h_t(p)\ge h_{t-1}(p)-6d$. Therefore,

$$
\begin{aligned}
C_0+C_1 &\ge 6(\phi(h_t(p)+d)-\phi(h_0(v)+4d))/5+2\max\{\phi(24jd),\phi(h_0(v)-d)\}\\
&\ge 6\phi(h_t(p)+d)/5\\
&\ge \phi(h_t(p))-\phi(h_{t-1}(p))+(\phi(h_t(p)+d)-\phi(h_t(p)))/10\\
&\ge D_1+D_2.
\end{aligned}
$$

(In the second step we use the inequality $2\max\{\phi(24jd), \phi(h_0(v) - d)\} \geq 6\phi(h_0(v) + 4d)/5$ which follows from Corollary A.1.1.)

We use a similar argument as above to handle subcase (b). The set X is the set of good tokens of v with heights in the interval $[e^u_{t-1}(v) - \rho, h_t(p) - d]$. Let c_x and C_1 be defined as in subcase (a). Let y denote $e^u_{t-1}(v) - \rho + 4d$. We first observe that since u sent a token to v during step t, $y \leq h_{t-1}(u) - 8d \leq h_{t-1}(p) - 7d$. Also, since p is a bad token we have $y \leq h_{t-1}(p) - 7d \leq h_t(p) - d$. As in subcase (a), we have:

$$\begin{aligned}
C_1 &\geq 6(\phi(h_t(p) + d) - \phi(y))/5 \\
&= (\phi(h_t(p) + d) - \phi(y)) + (\phi(h_t(p) + d) - \phi(y))/5 \\
&\geq \phi(h_t(p) + d) - \phi(h_{t-1}(p) + d) + (\phi(h_t(p) + d) - \phi(h_t(p)))/5 \\
&\geq \phi(h_t(p)) - \phi(h_{t-1}(p)) + D_2 \\
&= D_1 + D_2.
\end{aligned}$$

To complete the proof for case (ii), we show that for any token q of v, any incoming credit assigned by q to edge (u, v) that is used at step t for case (ii) is not used again for case (ii). To prove this, it is enough to note that if q belongs to X, for every further step $t' > t$ until q is transferred by u, we have $h_{t'}(q) \geq e^u_{t'-1}(v) - \rho$.

We need to consider case (iii) only under the assumption that $(u, v) \in E_t$. In this case we account for $D = (\phi(h_{t-1}(u) - d) - \phi(h_{t-1}(v) + d))/50$ units of potential. Again we consider two subcases depending on whether t is the least step in which (u, v) is live (subcase (a)) or not (subcase (b)). We first consider subcase (a). If $h_0(u) \geq h_{t-1}(u) - 12d$, then we use $C_0 = 2\max\{\phi(24jd), \phi(h_0(u) - d)\}$ units of the initial credit associated with (u, v). It follows from Corollary A.1.1 that $C_0 \geq \phi(h_{t-1}(u) - d)/50 \geq D$. If $h_0(u) < h_{t-1}(u) - 12d$, in addition to C_0, we also use part of the incoming credit associated with the set of tokens $Y = \{y : y \text{ is a token of } u \text{ and } h_0(u) < h_t(y) \leq h_{t-1}(u)\}$. Specifically, for every token y in Y, we use $(\phi(h_t(y) + d) - \phi(h_t(y)))/(50d)$ units of incoming credit that is assigned to (u, v) by y. (Note that since $h_t(y) > h_0(u)$, token y has moved and hence has some associated credit. Moreover, at least $(\phi(h_t(y) + d) - \phi(h_t(y)))/(20d)$ units of incoming credit remains unused in the analysis of case (ii).) Let this credit be denoted C_1. Thus, we obtain

$$\begin{aligned}
C_0 + C_1 &\geq C_0 + \left(\sum_{h_0(u) < k \leq h_{t-1}(u)} (\phi(k + d) - \phi(k))/(50d) \right) \\
&\geq C_0 + (\phi(h_{t-1}(u)) - \phi(h_0(u) + d))/50 \\
&\geq \phi(h_{t-1}(u))/50 \\
&\geq D.
\end{aligned}$$

(In the third step we use the inequality $C_0 \geq \phi(h_0(u) + d)/50$ which follows from Corollary A.1.1.)

We now consider subcase (b) of (iii). Since no token was sent along (u, v) in step t, we have $e^u_{t-1}(v) - \rho > h_{t-1}(u) - 12d$ ($\geq 24jd$). It follows that $e^u_{t-1}(v) - \rho > h_{t-1}(v) + 12d$. Since the last step in which (u, v) was live, at least $e^u_{t-1}(v) - \rho - h_{t-1}(v)$ tokens have left v. We use the outgoing credit C_2 assigned to (u, v) due to these token transmissions. We have

$$\begin{aligned}
C_2 &\geq \sum_{h_{t-1}(v) < k \leq e^u_{t-1}(v) - \rho} (\phi(k + d) - \phi(k))/(20d) \\
&\geq (\phi(e^u_{t-1}(v) - \rho) - \phi(h_{t-1}(v) + d))/20 \\
&\geq (\phi(e^u_{t-1}(v) - \rho + 11d) - \phi(h_{t-1}(v) + d))/50 \\
&\geq (\phi(h_{t-1}(u) - d) - \phi(h_{t-1}(v) + d))/50 \\
&= D.
\end{aligned}$$

(The third step follows from Lemma A.2. The fourth step follows from the inequality $e^u_{t-1}(v) - \rho > h_{t-1}(u) - 12d$.)

We note that the outgoing credit due to any such token x associated with edge (u, v) is used at most once in case (iii) because after step t, we have $e^u_t(v) - \rho \geq h_{t-1}(v)$. ■

A simple variant of **DS**, as suggested in [2], can be defined for the asynchronous network model in which the topology is fixed, but an adversary determines the speed at which each edge operates at every instant of time. An edge is said to be *live* for a unit interval of time, if every message that was in transit across the edge (in either direction) at the beginning of the interval is guaranteed to reach the end of the edge by the end of the interval. As shown in [2], the analysis for the dynamic synchronous case can be used for asynchronous networks to yield the same time bounds, under the assumption that in every unit interval of time the live edges of the network have edge expansion α.

6 Tight bounds on off-line load balancing

In this section, we analyze the load balancing problem in the off-line setting for both single-port and multi-port models. We derive nearly tight bounds on the minimum number of steps required to balance on arbitrary networks in terms of the node and edge expansion of the networks. We assume that the network is synchronous.

We first consider the network $G = (V, E)$ under the single-port model. For any subset X of V, let $\overline{X}$ denote $V \setminus X$, $m(X)$ denote the number of edges in a maximum matching between X and $\overline{X}$, $A(X)$ denote the set $\{v \in \overline{X} : \exists x \in X \text{ such that } (x, v) \in E\}$, and $B(X)$ denote the set $\{x \in X : \exists y \in A(X) \text{ such that } (x, y) \in E\}$. For subsets X and Y of V, let $M(X, Y)$ denote the set of edges with one endpoint in X and the other in Y.

Lemma 6.1 *For any network $G = (V, E)$ with node expansion μ and any subset X of V, we have $m(X) \geq \mu \min\{|X|, |\overline{X}|\}/(1 + \mu)$. Moreover, there exists a set X of size at most $(1 + \mu)|V|/2$ such that $m(X) \leq \mu|X|/(1 + \mu)$.*

Proof: Without loss of generality, assume that $|X| \leq |\overline{X}|$. Consider the bipartite graph $H = (B(X), A(X), M(X, \overline{X}))$. A maximum matching in H is equal to a maximum flow in the graph $I = (B(X) \cup A(X) \cup \{s, t\}, M(X, \overline{X}) \cup \{(s, x) : x \in B(X)\} \cup \{(x, t) : x \in A(X)\})$ from source s to sink t. (All of the edges of I have unit capacity.) We will show that every cut C of I separating s and t is of cardinality at least $\mu|X|/(1 + \mu)$.

Consider any cut $C = (S, T)$ with $s \in S$ and $t \in T$. Let $Y = T \cap B(X)$ and $Z = T \cap A(X)$. The cardinality of C is $|Y| + |M(Y, A(X) \setminus Z)| + |M(B(X) \setminus Y, Z)| + |A(X) \setminus Z| \geq |Y| + |M(B(X) \setminus Y, Z)| + |A(X) \setminus Z| \geq |A(X \setminus Y)|$. Since $|C| \geq |Y|$, and $|A(X \setminus Y)| \geq \mu|X \setminus Y|$, we have $|C| \geq \mu|X|/(1 + \mu)$.

For the second part of the lemma, consider set Y of size at most $|V|/2$ such that $A(Y) = \mu|Y|$. If we set $X = Y \cup A(Y)$, then $m(X) \leq A(Y) = \mu|X|/(1 + \mu)$. ■

Theorem 1 of [28] obtains tight bounds on the offline complexity of load balancing in terms of the function m. By invoking Lemma 6.1, we obtain

Lemma 6.2 ([28]) *Any network G with node expansion μ and initial imbalance Δ can be balanced in at most $\lceil \Delta(1 + \mu)/\mu \rceil$ steps so that every node has at most $\lceil \rho \rceil + 1$ tokens.. Moreover, there exists a network G and an initial load distribution with imbalance Δ such that any algorithm takes at least $\lceil \Delta(1 + \mu)/\mu \rceil$ steps to balance G such that every node has at most $\lceil \rho \rceil$ tokens.* ■

By using the techniques of [28], we can modify the proof of Lemma 6.2 to show that any network G with node expansion μ and initial imbalance Δ can be globally balanced to within 3 tokens in at most $2\lceil \Delta(1+\mu)/\mu \rceil$ steps. Lemma 6.2 implies that our local single-port algorithm is not optimal for all networks.

However, there exists a class of graphs, e.g., constant-degree expanders, for which the local algorithm is optimal. A network for which the local algorithm is not optimal is the hypercube. The local algorithm balances in $\Omega(\Delta \log n)$ time, while there exists an $O(\Delta\sqrt{\log n} + \log^2 n)$ time load balancing algorithm for the hypercube [33] which is optimal for Δ sufficiently large.

The proof of Lemma 6.2 can be modified to establish the following result for the multi-port model.

Lemma 6.3 *Any network G with edge expansion α and initial imbalance Δ can balanced in at most $\lceil \Delta/\alpha \rceil$ steps so that every node has at most $\lceil \rho \rceil + d$ tokens. Moreover, for every network G, there exists an initial load distribution with imbalance Δ such that any algorithm takes at least $\lceil \Delta/\alpha \rceil$ steps to balance G so that every node has at most $\lceil \rho \rceil$ tokens.*

Proof Sketch: We prove that there exists an off-line algorithm that balances to within d tokens in at most $T = \max_{\emptyset \subset X \subset V} \lceil \frac{|I(X) - \rho|X||}{|M(X,\overline{X})|} \rceil$ steps, where $I(X)$ is the number of tokens held by nodes in X in the initial distribution. For all $X \subseteq V$, we have (i) $|I(X) - \rho|X|| \leq \Delta \min\{|X|, |\overline{X}|\}$ and (ii) $|M(X, \overline{X})| \geq \alpha \min\{|X|, |\overline{X}|\}$. It follows from (i) and (ii) that $T \leq \lceil \Delta/\alpha \rceil$.

We essentially modify the proofs of Theorem 1 and Lemma 4 of [28] (where the single-port model was assumed) to establish the desired claims for the multi-port model. We transform the load balancing problem on G to a network flow problem on a directed graph $H = (V', E')$ which is constructed as follows. Let V_i be $\{\langle v, i \rangle : v \in V\}$, $0 \leq i \leq T$. Let E_i be $\{(\langle u, i \rangle, \langle v, i+1 \rangle) : (u,v) \in E \text{ or } u = v\}$, $0 \leq i < T$. We set V' to $\{s\} \cup \bigcup_{0 \leq i \leq T} V_i \cup \{t\}$, and E' to $\{(s, \langle v, 0 \rangle) : v \in V\} \cup \bigcup_{0 \leq i < T} E_i \cup \{(\langle v, T \rangle, t) : v \in V\}$. For any v in V, the capacity of the edge $(s, \langle v, 0 \rangle)$ is $w(v)$. For any (u,v) in E, the capacity of any edge $(\langle u, i \rangle, \langle v, i+1 \rangle)$, $0 \leq i < T$, is 1. For any v in V, the capacity of any edge $(\langle v, i \rangle, \langle v, i+1 \rangle)$, $0 \leq i < T$, is ∞. For any v in V, the capacity of the edge $(\langle v, T \rangle, t)$ is $\lceil \rho \rceil + d$.

We show that the value of the maximum integral flow in H is equal to the total number of tokens N in V from which it follows that there exists an off-line algorithm that balances to within d tokens in T steps. Consider any cut $C = (S, T)$ of H separating $s \in S$ and $t \in T$. Let $S_i = S \cap V_i$ and $D(S_i) = \{v \in V : \langle v, i \rangle \in S_i\}$. If $S_0 = \emptyset$, or $S_T = V_T$, or there is an edge of infinite capacity then the capacity of C is at least N. Otherwise, the number of edges from V_i to V_{i+1} that belong to the cut is at least $|M(D(S_i), \overline{D(S_i)})| - d(|S_{i+1}| - |S_i|)$. Thus the capacity of C is at least

$$\sum_{v \in V_0 \setminus S_0} w(v) + \sum_{i=0}^{T-1} \Big(|M(D(S_i), \overline{D(S_i)})| - d(|S_{i+1}| - |S_i|) \Big) + (\lceil \rho \rceil + d)|S_T| \geq N.$$

Since capacity of the cut $(\{s\}, V' \setminus \{s\})$ equals N, the maximum flow in H is N.

To prove the second part of the lemma, given any network G with a partition (V_1, V_2) of its nodes such that $|V_1| \leq n/2$ and $|M(V_1, V_2)| = \alpha|V_1|$, we define an initial load distribution in which each node in V_1 has Δ tokens and each node in V_2 has zero tokens. It is easily verified that a proper choice of Δ establishes the desired claim. ■

Lemma 6.3 implies that the local multi-port algorithm is asymptotically optimal for *all* networks. As in the single-port case, we can modify the above proof to obtain upper bounds on the off-line complexity of globally balancing a network. We can show that any network G with edge expansion α and initial imbalance Δ can globally balanced to within $d + 1$ tokens in at most $2\lceil \Delta/\alpha \rceil$ steps.

References

[1] Y. Afek, E. Gafni, and A. Rosen. The slide mechanism with applications in dynamic networks. In *Proceedings of the 11th ACM Symposium on Principles of Distributed Computing*, pages 35–46, August 1992.

[2] W. Aiello, B. Awerbuch, B. Maggs, and S. Rao. Approximate load balancing on dynamic and asynchronous networks. In *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing*, pages 632–641, May 1993.

[3] M. Ajtai, J. Komlós, and E. Szemerédi. Sorting in $c \log n$ parallel steps. *Combinatorica*, 3:1–19, 1983.

[4] J. Aspnes, M. Herlihy, and N. Shavit. Counting networks and multiprocessor co-ordination. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, pages 348–358, May 1991.

[5] B. Awerbuch and T. Leighton. A simple local-control approximation algorithm for multi-commodity flow. In *Proceedings of the 34th Annual Symposium on Foundations of Computer Science*, pages 459–468, October 1993.

[6] B. Awerbuch and T. Leighton. Improved approximation algorithms for the multi-commodity flow problem and local competitive routing in dynamic networks. In *Proceedings of the 26th Annual ACM Symposium on the Theory of Computing*, pages 487–496, May 1994.

[7] B. Awerbuch, Y. Mansour, and N. Shavit. End-to-end communication with polynomial overhead. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pages 358–363, October 1989.

[8] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods.* Chapter 7, Prentice–Hall, Englewood Cliffs, NJ, 1989.

[9] A. Broder, A. M. Frieze, E. Shamir, and E. Upfal. Near–perfect token distribution. In *Proceedings of the 19th International Colloquium on Automata, Languages and Programming*, pages 308–317, July 1992.

[10] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–507, 1952.

[11] Y. C. Chow and W. Kohler. Models for dynamic load balancing in a heterogeneous multiple processor system. *IEEE Transactions on Computers*, C–28(5):57–68, November 1980.

[12] G. Cybenko. Dynamic load balancing for distributed memory multiprocessors. *Journal of Parallel and Distributed Computing*, 7(2):279–301, 1989.

[13] D. Eager, E. Lazowska, and J. Zahorjan. Adaptive load sharing in homogeneous distributed systems. *IEEE Transactions on Software Engineering*, SE–12(5):662–675, 1986.

[14] B. Ghosh and S. Muthukrishnan. Dynamic load balancing on parallel and distributed networks by random matchings. In *Proceedings of the 6th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 226–235, June 1994.

[15] B. Goldberg and P. Hudak. Implementing functional programs on a hypercube multiprocessor. In *Proceedings of the 4th Conference on Hypercubes, Concurrent Computers and Applications*, volume 1, pages 489–503, 1989.

[16] A. Heirich and S. Taylor. A parabolic theory of load balance. Research Report Caltech-CS-TR-93-25, Caltech Scalable Concurrent Computation Lab, Pasadena, CA, March 1993.

[17] K. T. Herley. A note on the token distribution problem. *Information Processing Letters*, 28:329–334, 1991.

[18] S. H. Hosseini, B. Litow, M. Malkawi, J. McPherson, and K. Vairavan. Analysis of a graph coloring based distributed load balancing algorithm. *Journal of Parallel and Distributed Computing*, 10(2):160–166, 1990.

[19] J. JáJá and K. W. Ryu. Load balancing and routing on the hypercube and related networks. *Journal of Parallel and Distributed Computing*, 14(4):431–435, 1992.

[20] M. R. Jerrum and A. Sinclair. Conductance and the rapid mixing property for Markov chains: the approximation of the permanent resolved. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 235–244, May 1988.

[21] R. Karp and Y. Zhang. A randomized parallel branch-and-bound procedure. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 290–300, May 1988.

[22] M. Klugerman and C. G. Plaxton. Small depth counting networks. In *Proceedings of the 24th Annual ACM Symposium on the Theory of Computing*, pages 417–428, May 1992.

[23] T. Leighton, C. E. Leiserson, and D. Kravets. Theory of parallel and VLSI computation. Research Seminar Series Report MIT/LCS/RSS 8, MIT Laboratory for Computer Science, May 1990.

[24] F. C. H. Lin and R. M. Keller. The gradient model load balancing method. *IEEE Transactions on Software Engineering*, SE–13(1):32–38, 1987.

[25] R. Lüling and B. Monien. Load balancing for distributed branch and bound algorithms. In *Proceedings of the 6th International Parallel Processing Symposium*, pages 543–549, March 1992.

[26] R. Lüling and B. Monien. A dynamic distributed load balancing algorithm with provable good performance. In *Proceedings of the 5th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 164–172, June 1993.

[27] R. Lüling, B. Monien, and F. Ramme. Load balancing in large networks: A comparative study. In *Proceedings of the 3rd IEEE Symposium on Parallel and Distributed Processing*, pages 686–689, December 1991.

[28] F. Meyer auf der Heide, B. Osterdiekhoff, and R. Wanka. Strongly adaptive token distribution. In *Proceedings of the 20th International Colloquium on Automata, Languages and Programming*, pages 398–409, July 1993.

[29] M. Mihail. Conductance and convergence of Markov chains - a combinatorial treatment of expanders. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pages 526–531, October 1989.

[30] L. M. Ni, C. Xu, and T. B. Gendreau. Distributed drafting algorithm for load balancing. *IEEE Transactions on Software Engineering*, SE–11(10):1153–1161, October 1985.

[31] D. Peleg and E. Upfal. The generalized packet routing problem. *Theoretical Computer Science*, 53:281–293, 1987.

[32] D. Peleg and E. Upfal. The token distribution problem. *SIAM Journal on Computing*, 18:229–243, 1989.

[33] C. G. Plaxton. Load balancing, selection and sorting on the hypercube. In *Proceedings of the 1989 ACM Symposium on Parallel Algorithms and Architectures*, pages 64–73, June 1989.

[34] J. Stankovic. Simulations of three adaptive, decentralized controlled, job scheduling algorithms. *Computer Networks*, 8:199–217, 1984.

[35] R. Subramanian and I. D. Scherson. An analysis of diffusive load balancing. In *Proceedings of the 1994 ACM Symposium on Parallel Algorithms and Architectures*, pages 220–225, June 1994.

[36] A. N. Tantawi and D. Towsley. Optimal static load balancing in distributed computer systems. *Journal of the ACM*, 32:445–465, April 1985.

[37] E. Upfal. An $O(\log N)$ deterministic packet routing scheme. pages 241–250, May 1989.

[38] R. D. Williams. Performance of dynamic load balancing algorithms for unstructured mesh calculations. *Concurrency: Practice and Experience*, 3(5):457–481, 1991.

A Some Inequalities

Let ν equal $\alpha/(cd^2)$. For the following we set c large enough so that $(1+\nu)^{12d} \leq 3/2$.

Lemma A.1 *For any integer x, if $\phi(x) > 0$, then $\phi(x+12d) \leq 3\phi(x)/2$.*

Proof: Since $\phi(x) > 0$, we have

$$\begin{aligned} \phi(x+12d) &= (1+\nu)^{12d}\phi(x) \\ &\leq 3\phi(x)/2. \end{aligned}$$

■

Corollary A.1.1 *For any integer x we have*

$$\max\{\phi(24jd), \phi(x-12d)\} \geq 2\phi(x)/3$$

Lemma A.2 *For any integer x and y, if $\phi(x) > 0$ and $x - y \geq 11d$, then we have $\phi(x) - \phi(y) \geq 2(\phi(x+11d) - \phi(y))/5$.*

Proof:

$$\begin{aligned} 2(\phi(x+11d) - \phi(y))/5 &= 2(\phi(x+11d) - \phi(x))/5 \\ &\quad +2(\phi(x) - \phi(y))/5 \\ &\leq 2(1+\nu)^{11d}(\phi(x) - \phi(x-11d))/5 \\ &\quad +2(\phi(x) - \phi(y))/5 \\ &\leq 2(1+\nu)^{11d}(\phi(x) - \phi(y))/5 \\ &\quad +2(\phi(x) - \phi(y))/5 \\ &\leq \phi(x) - \phi(y). \end{aligned}$$

(In the second step we use the inequality $x - 11d \geq y$. In the last step we use the inequality $(1+\nu)^{11d} \leq 3/2$.)

■

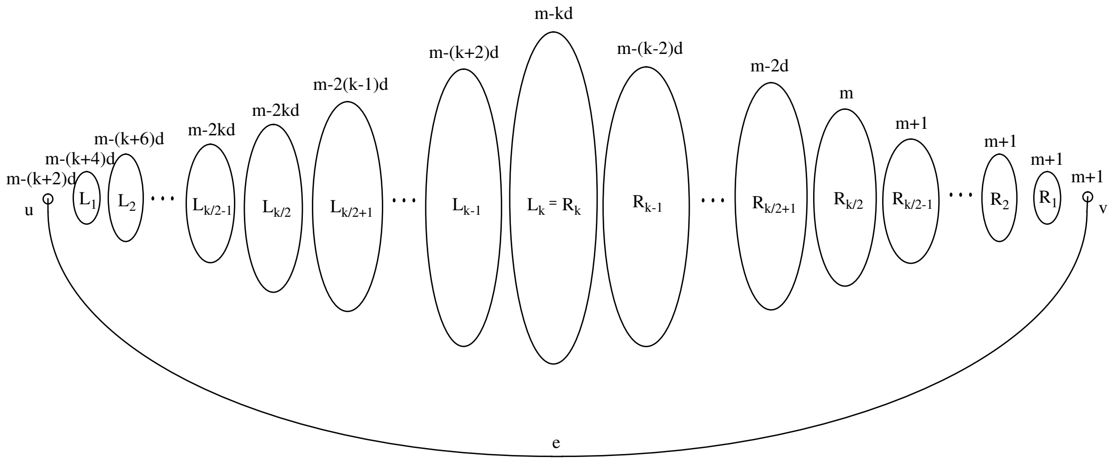

Figure 1: The initial tokens distribution on G for the first case.

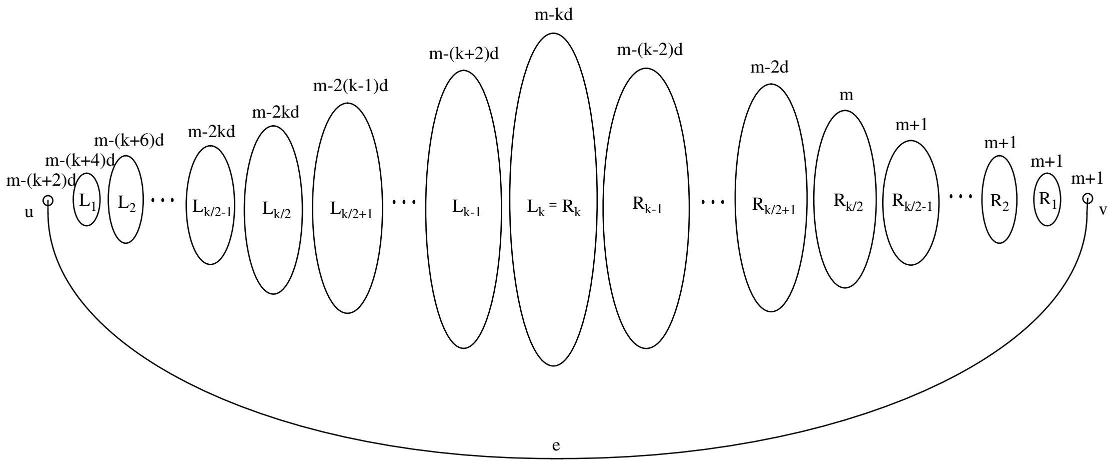

Figure 2: The initial tokens distribution on G for the second case.