

Approximate Majorization and Fair Online Load Balancing

Ashish Goel *

Adam Meyerson †

Serge Plotkin ‡

July 7, 2000

Abstract

This paper relates the notion of fairness in online routing and load balancing to *vector majorization* as developed by Hardy, Littlewood, and Polya [9]. We define α -*supermajorization* as an approximate form of vector majorization, and show that this definition generalizes and strengthens the prefix measure proposed by Kleinberg, Rabani and Tardos [11] as well as the popular notion of *max-min fairness*.

The paper revisits the problem of online load-balancing for unrelated $1-\infty$ machines from the viewpoint of fairness. We prove that a greedy approach is $O(\log n)$ -supermajorized by all other allocations, where n is the number of jobs. This means the greedy approach is globally $O(\log n)$ -*fair*. This may be contrasted with polynomial lower bounds presented in [7] for fair online routing.

We also define a machine-centric view of fairness using the related concept of *submajorization*. We prove that the greedy online algorithm is globally $O(\log m)$ -*balanced*, where m is the number of machines.

*Department of Computer Science, University of Southern California, Los Angeles CA 90089-0781. Email: agoel@cs.usc.edu.

†Department of Computer Science, Stanford University CA 94305. Email: awm@cs.stanford.edu.

‡Supported by ARO Grants DAAG55-98-1-0170 and ONR Grant N00014-98-1-0589. Department of Computer Science, Stanford University CA 94305. Email: plotkin@cs.stanford.edu.

1 Introduction

Fair Resource Allocation The problems of online allocation of resources in order to minimize load [6, 3] or maximize profit [4] have been well-studied. In a real system, any allocation must also guarantee some form of fairness. Starving one job of resources in order to increase the allocation to another is not always acceptable. Intuitively, if particular jobs demand conflicting resources, those resources should be allocated in an egalitarian manner.

Before addressing this issue, we must formalize our intuitive notion of fairness. Several candidate definitions have been proposed. The “greatest good for the greatest numbers” criterion is equivalent to maximizing the total bandwidth allocated. This criterion optimizes for the overall usage of resources, but may starve a job which requires many resources in favor of several jobs which require few. Another criterion maximizes the bandwidth allocated to the job which receives least bandwidth. For the case of online load balancing, this was considered by [6, 3] and a greedy allocation was proven to be $O(\log m)$ competitive. However, if two jobs do not conflict (demanding distinct resources), then a guarantee on the minimum assigned bandwidth does not capture our intuition that these jobs should be essentially independent. Yet another criterion is max-min fairness [1, 2, 5]. Max-min fairness makes sense when the assignment of resources is fixed and we need to determine the split of each resource between the jobs requiring it; however, max-min fairness is poorly defined in the scenario where resource assignment must be performed online [7]. Recently, Kleinberg, Rabani, and Tardos [11] proposed a measure of *global fairness* which unifies all the above criteria. We will extend and strengthen their definition by relating it to the concept of *vector majorization* defined by Hardy, Littlewood, and Polya [9].

We compute a vector representing the amount of resources assigned to each request. This *bandwidth vector* is arranged in nondecreasing order of resources. Given two such vectors X and Y , X is majorized by Y ($X \prec Y$) if the sum of the first k terms of X is at least the sum of the first k terms of Y for every k , and the total sums of the vectors are equal. Conceptually, if X is majorized by Y , then assignment X gives more bandwidth to the requests receiving least, and is therefore more fair. We extend majorization by defining α -supermajorization; X is α -supermajorized by Y if (for every k), α times the sum of the first k terms of X is at least equal to the sum of the first k terms of Y . We present several useful theorems regarding supermajorization which follow directly from the work of Hardy, Littlewood, and Polya [9] on majorization.

We consider an allocation of resources X to be α -fair if every other valid allocation Y satisfies $X \prec^\alpha Y$ (Y α -supermajorizes X). If there exists an allocation which is 1-fair, then this allocation is identical to the globally fair allocation defined by Kleinberg, Rabani, and Tardos [11] and also satisfies the property of being *max-min fair*. We believe that the concept of approximate supermajorization will find other important applications in fair resource allocation problems.

After presenting our formal definition of fairness, the remainder of the paper is devoted to the “one-infinity” model of online machine scheduling [6, 3] where each job will run on one of a specified subset of machines. The greedy algorithm for this problem is a slight restatement of Graham’s rule [8] – assign each job to the least loaded machine capable of processing the job. We prove that this greedy algorithm obtains a globally $O(\log n)$ -fair allocation of resources to the jobs. This may be contrasted with the polynomial lower bounds presented for the more difficult routing problem in [7]. Alternately, we can view fairness from the standpoint of the machines. We show that the loads of machines are $O(\log m)$ -balanced. This strengthens previous results by showing that online greedy allocations are close to fair throughout, rather than just at the “endpoint” of

the minimum-bandwidth job (or maximum-loaded machine).

Lower bounds on the bandwidth assigned to the least-bandwidth job directly translate to lower bounds on global fairness. It follows that any online algorithm must be globally $\Omega(\log m)$ -fair as well as globally $\Omega(\log m)$ -balanced [6, 3].

Related work: Kleinberg *et al.* [11] addressed the problem of fair resource allocation in the *offline* setting, and gave a polynomial-time algorithm which computes a globally fair solution given all the requests up-front. This solution is also globally 1-fair by our definition. Goel, Meyerson, and Plotkin [7] gave a globally polylogarithmic-fair online algorithm for the more general problem of routing in communication networks under the assumption that the routing algorithm can exercise some control over the bandwidth allocation.

2 Measuring Fairness by Majorization

For any vector $a = \langle a_1, a_2, \dots, a_n \rangle$, denote the i -th smallest component of a by $a_{(i)}$. Suppose we are given two n -dimensional vectors x and y with non-negative components.

Definition 1 *Given two n -dimensional vectors x and y , x is said to be majorized by y (y majorizes x) if each of the following is true:*

1. $\sum_{i=1}^k x_{(i)} \geq \sum_{i=1}^k y_{(i)}$, for all $1 \leq k \leq n$
2. $\sum_{i=1}^n x_{(i)} = \sum_{i=1}^n y_{(i)}$

We use the notation $x \prec y$ to denote the above relation.

In this paper we will only concern ourselves with vectors where each component is non-negative. Notice that majorization is a property of the multisets $\{x_1, x_2, \dots, x_n\}$ and $\{y_1, \dots, y_n\}$. However it is advantageous to think of them as vectors. Majorization was first studied by Hardy, Littlewood, and Polya [9]. They proved that $\sum_i g(x_i) \leq \sum_i g(y_i)$ for all convex functions g iff $x \prec y$. Another interpretation of this relation is that $x \prec y$ if and only if x is a fairer (than y) allocation of a fixed resource. For a detailed exposition of majorization, see [13]. The concept of majorization was first used in the context of fairness by Lorenz [12].

Hardy, Littlewood, and Polya [10] also proved the following:

Theorem 2.1 *$x \prec y$ iff $x = yP$ for a doubly stochastic matrix P .*

Recall that a doubly stochastic matrix is a matrix in which all rows and columns sum to 1. Similarly, a doubly superstochastic matrix has all rows and columns summing to *at least* 1. Hardy *et al.* define x to be *supermajorized* by y iff $x = yP$ for some doubly superstochastic matrix P .

A concept very similar to supermajorization in the context of fair bandwidth allocation was rediscovered (before our work) by Kleinberg, Rabani, and Tardos [11]. A vector of bandwidth allocations x is said to be α -prefix-competitive with another vector y if $\alpha \sum_{i=1}^k x_{(i)} \geq \sum_{i=1}^k y_{(i)}$. Also, x is said to be coordinate-wise α -competitive to y if $\alpha x_{(i)} \geq y_{(i)}$. They gave an elegant offline algorithm that obtains a bandwidth allocation that is supermajorized by (i.e. is 1-prefix-competitive

to) any other allocation. They further present a solution to the single source unsplittable flow problem that is 2-prefix-competitive to any other solution.

Definition 2 A globally fair solution is an assignment of jobs to machines which produces a bandwidth allocation that is supermajorized by any other feasible bandwidth allocation.

Global fairness is a very strong condition, and cannot be achieved in the context of online load balancing. In fact, it is not clear a priori that this property can be achieved even in the offline case – hence, the result of Kleinberg, Rabani, and Tardos is quite striking. To capture approximate fairness, we introduce α -supermajorization which is a slight variation of the notion of supermajorization defined by Hardy, Littlewood, and Polya.

Definition 3 Given two n -dimensional vectors x and y , x is said to be α -supermajorized by y if $\alpha \sum_{i=1}^k x_{(i)} \geq \sum_{i=1}^k y_{(i)}$ for all $k \leq n$. This is denoted by $x \prec^\alpha y$.

The following properties of supermajorization are immediate from the above definitions.

Theorem 2.2 1. $x \prec^\alpha y$ iff $\alpha x = yP$ for some doubly superstochastic matrix P .

2. $x \prec y \Rightarrow x \prec^1 y$

3. $x \prec^\alpha y, y \prec^\beta z \Rightarrow x \prec^{\alpha\beta} z$.

Approximate submajorization can be similarly defined.

Definition 4 Given two n -dimensional vectors x and y , x is said to be α -submajorized by y if $\sum_{i=1}^k x_{(n+1-i)} \leq \alpha \sum_{i=1}^k y_{(n+1-i)}$ for all $k \leq n$. This is denoted by $x \prec_\alpha y$.

Definition 5 A vector is said to be globally α -fair if it is α -supermajorized by any other feasible vector. A vector is said to be globally α -balanced if it is α -submajorized by any other feasible vector.

In the case of load balancing, global α -fairness will be used for a job-centric view of fairness: the k poorest jobs together in a globally α -fair solution must be at least $1/\alpha$ times as rich as the k poorest jobs in any other allocation. Global α -balance will represent a machine-centric view of fairness: the k most loaded machines in a globally α -balanced solution can together have a load that is at most α times the load of the k most loaded machines in any other allocation. Figure 1 illustrates the difference between approximate submajorization and supermajorization.

2.1 Fairness in One-Infinity Load Balancing

We are given a set of m machines. Jobs arrive one at a time online. Each job specifies some subset of the machines on which it is allowed to run. We must select a machine from the stated subset on which to assign the job. The total number of jobs is n .

Each job assigned to machine i receives *bandwidth* equal to $1/L_i$ where L_i is the total number of jobs assigned to machine i .

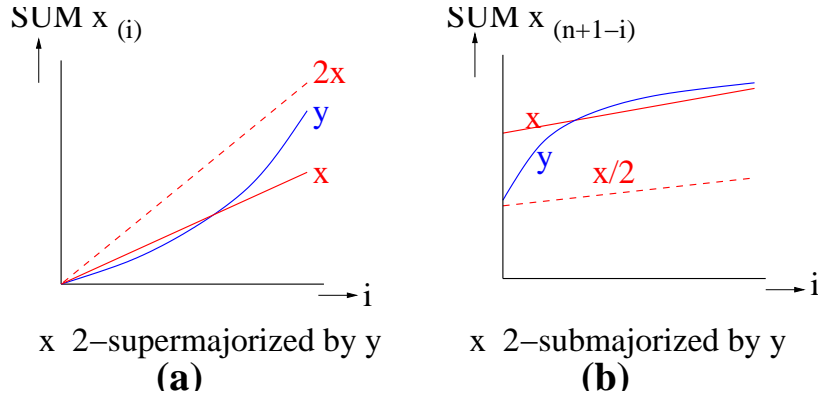


Figure 1: Approximate supermajorization (a) captures fairness when x and y are allocations of resources to jobs, while approximate submajorization (b) captures fairness when x and y are loads on machines. In both cases above, x is at least half as fair/balanced as y .

Definition 6 *The bandwidth vector for an assignment of jobs to machines is composed of the jobs' bandwidths listed in nondecreasing order.*

Definition 7 *The load vector for an assignment of jobs to machines is composed of the machines' loads (L_i).*

Definition 8 *A solution S is α coordinate-wise competitive if and only if $\alpha S_{(i)} \geq T_{(i)}$ for every coordinate i and every legal solution T .*

We observe that coordinate-wise competitiveness is stronger than supermajorization. In other words, if S is α coordinate-wise competitive, it immediately follows that S is also globally α -fair. Note that the reverse implication does not hold (S may be globally α -fair but not coordinate-wise competitive).

Our algorithm for $1-\infty$ load balancing will provide a solution which has a globally $O(\log n)$ -fair bandwidth vector and a globally $O(\log m)$ -balanced load vector.

3 The Algorithm

We will use a simple greedy algorithm. Whenever a job comes in, we find the least loaded of the machines where the job can run, and assign the job to that machine. Ties are broken in some arbitrary way (say by assigning the job to the lowest numbered of the tied machines).

Remark 3.1 *One can note that if the optimally fair algorithm places t jobs on a specific machine, then the online cannot place more than $(t+1)O(\log m)$ jobs on this machine. This follows from the fact that any job which the optimum placed on a machine with at least $t+2$ jobs cannot be legally placed on a machine on which the optimum placed only t . It is now straightforward to prove that there exists an allocation of bandwidths by which the greedy algorithm is globally $O(\log m \log n)$ -fair. Using majorization techniques, it can be shown that this competitive ratio holds even if bandwidths are divided equally between jobs on the same machine. However, we will instead present a more complex proof to improve the result to global $O(\log n)$ -fairness.*

We will first prove a straightforward claim about the number of jobs placed currently on a machine when a new job is scheduled there. We then show that assigning arbitrary bandwidths to jobs (instead of an equal split) cannot help make the solution more fair. We conclude by showing coordinate-wise bandwidth competitiveness against global fairness using assigned bandwidths, from which global approximate-fairness without assigned bandwidths will follow.

3.1 Job Placement

Define R_i to be the set of jobs which the online greedy algorithm placed on machines which had at least i jobs on them already. Suppose for some subset R_i^b of R_i , there is a way to schedule all the jobs in R_i^b without exceeding b per machine. The following is a standard result for load balancing.

Theorem 3.1 *The online algorithm places at least $(1/2)|R_i^b|$ jobs on machines which have between i and $i + b$ jobs at the time they're placed.*

Proof: Consider subset S of R_i^b , where S is the set of jobs which the online placed on machines with at least $i + b$ jobs already present. Assign each member of S to the machine it would be placed in had we scheduled all the jobs in R_i^b without exceeding b per machine. At the time each job in S arrived, it was placed on a machine with at least $i + b$ jobs. Therefore its “assigned” machine had at least $i + b$ jobs. At most b jobs from S are assigned to each machine, and each machine with any jobs from S assigned to it has at least $i + b$ jobs. It follows that at least $|S|$ jobs were placed “between” i and $i + b$, so at least $(1/2)|R_i^b|$ jobs must have been placed on machines which had, at the time of placement, between i and $i + b$ jobs. ■

3.2 Bandwidth Assignment

Suppose we could assign any bandwidth we like to the jobs, provided only that the total bandwidths of jobs on each machine is at most one. We prove that any globally α -fair solution which can assign arbitrary bandwidth to jobs will remain at least as fair if the jobs split the machines equally instead.

Theorem 3.2 *Suppose there exists an assignment S of bandwidths to jobs and jobs to machines. If we define S' to assign jobs in the same way while splitting bandwidth equally among jobs sent to the same machine, then $S' \prec^1 S$.*

Proof: Consider reassigning the bandwidths to jobs on a certain machine, so that each job receives the average bandwidth. This is equivalent to multiplying the vector S by a doubly stochastic matrix with $1/j$ in the positions corresponding to pairs of the j jobs on the certain machine and the identity matrix elsewhere. Since the product of doubly stochastic matrices is doubly stochastic, we apply theorem 2.1 to show that $S' \prec^1 S$. ■

3.3 Global $O(\log n)$ -fairness

We will compare our online solution to some solution T' . We prove that, for any T' which splits the bandwidth of every machine in an equal fashion, we can assign bandwidths to the jobs without

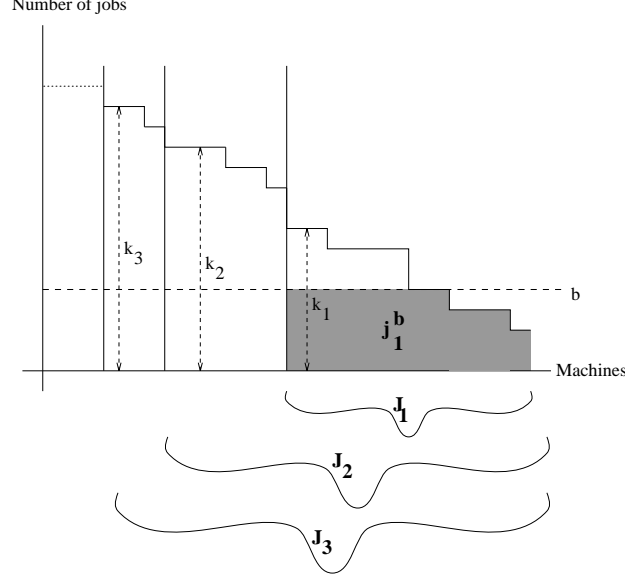


Figure 2: The machine loads for the schedule T'

exceeding one unit of bandwidth per machine, such that our bandwidth vector is coordinate-wise within $O(\log n)$ of the bandwidth vector of T' . It follows that our solution is $O(\log n)$ -supermajorized by T' . By applying theorem 3.2, we proceed to show that even if we were to divide bandwidths equally (eliminating the need for prior knowledge of T'), our solution would still be $O(\log n)$ -supermajorized by T' . From this it follows that our solution is $O(\log n)$ -fair.

We define m to be the number of machines and t to be the maximum number of jobs which solution T' places on any machine. If n is the total number of jobs, we can guarantee that $t \leq n$ and $m \leq n$ since allowing machines on which T' places zero jobs can only help our algorithm.

We define the set J_i (see figure 2 for an illustration) of machines to consist of all the machines except the $m(1/2)^i$ to which solution T' (to which we are comparing) assigns most jobs. Notice that J_0 is empty and $J_{1+\log m}$ is all the machines. We define k_i to be the maximum number of jobs on a machine in J_i . It follows that $0 = k_0 \leq k_1 \leq k_2 \dots \leq k_{1+\log m} = t$.

Define j_i^b to be the number of jobs which solution T' places on machines in J_i , counting at most b jobs per machine (i.e. sum of the min of b and the machine load for each machine).

Define r_i^b to be the total jobs “under” $b + \sum_{x=1}^{i-1} k_x$ in the online solution S (i.e. the sum over machines of the min of the online load of the machine and $b + \sum_{x=1}^{i-1} k_x$).

Lemma 3.1 *For every $i \geq 1$ and $b \geq 0$ we have $r_i^b \geq j_i^b$.*

Proof: The proof will be by induction on i .

Base Case: $i = 1$. Exactly half the machines are in J_1 . We pair each machine in J_1 with a machine not in J_1 . The machine not in J_1 always has at least as many jobs as the machine in J_1 , because of the definition of the set. Thus the total number of jobs “below” b in solution T' is at least $2j_1^b$. Solution T scheduled at least $2j_1^b$ jobs using b capacity per machine, so the online will schedule at least half that many “under” b (see theorem 3.1). It follows that $r_1^b \geq j_1^b$ as desired.

Inductive step. We suppose $b \leq k_i$. Since every machine not in J_i has at least k_i jobs on it,

there are exactly $mb(1/2)^i$ jobs below b and not in J_i . Overall, solution T' must place $j_i^b + mb(1/2)^i$ jobs below b . The online solution schedules $r_{i-1}^{k_{i-1}}$ jobs below $k_1 + k_2 + \dots + k_{i-1}$. Thus at least $j_i^b - r_{i-1}^{k_{i-1}} + mb(1/2)^i$ jobs were scheduled below b in T' but above $k_1 + k_2 + \dots + k_{i-1}$ in S . In the next b capacity, we will schedule at least half of them (see theorem 3.1). This means S schedules at least

$$r_i^b \geq r_{i-1}^{k_{i-1}} + \frac{1}{2}(j_i^b - r_{i-1}^{k_{i-1}} + \frac{mb}{2^i}) \geq \frac{1}{2}(j_i^b + r_{i-1}^{k_{i-1}} + \frac{mb}{2^i})$$

jobs below $b + k_1 + k_2 + \dots + k_{i-1}$.

Using the inductive hypothesis, we know that $r_{i-1}^{k_{i-1}} \geq j_{i-1}^{k_{i-1}}$.

Each machine in J_{i-1} has at most k_{i-1} jobs on it, so it follows that $j_{i-1}^{k_{i-1}}$ represents all the jobs on machines in J_{i-1} . It follows that $j_{i-1}^{k_{i-1}} \geq j_{i-1}^b$.

There are $m(1/2)^i$ machines in $J_i - J_{i-1}$, and each contributes at most b jobs below b , so we know $j_i^b \leq j_{i-1}^b + mb(1/2)^i$.

It follows that $r_{i-1}^{k_{i-1}} + mb(1/2)^i \geq j_i^b$.

Thus we can write the following.

$$r_i^b \geq \frac{1}{2}(j_i^b + r_{i-1}^{k_{i-1}} + \frac{mb}{2^i}) \geq \frac{1}{2}(j_i^b + j_i^b) \geq j_i^b$$

which proves the claim for $b \leq k_i$. On the other hand, if $b > k_i$ then j_i^b represents all jobs on machines in J_i and it follows that $r_i^b \geq r_i^{k_i} \geq j_i^{k_i} = j_i^b$. ■

We assign bandwidth to jobs in the online solution S . We define $\alpha = \log m + \log t$. Suppose a job is the a th job on the machine it's running on. There is some unique choice of i and b such that $a = k_1 + k_2 + \dots + k_i + b$ and $b \leq k_{i+1}$. The job receives bandwidth equal to the minimum of $1/\alpha k_i$ and $1/\alpha b$. If $i = 0$, the bandwidth assigned is $1/\alpha b$. We will first show that this allocation is coordinate-wise competitive, then use theorem 3.2 to complete the proof of global $O(\log n)$ -fairness.

Lemma 3.2 *For any coordinate j , $\alpha S_{(j)} \geq T'_{(j)}$.*

Proof: Let $b = 1/T'_{(j)}$. Let i be the maximum integer such that $k_i \leq b$. Any machine not in J_{i+1} has at least $k_{i+1} > b$ jobs on it. Therefore the jobs on this machine receive less than $1/b$ bandwidth in solution T' . It follows that all jobs receiving $1/b$ are on machines with at most b jobs, all of which are in J_{i+1} . Thus at most j_{i+1}^b jobs receive $1/b$ bandwidth or more in solution T' . We conclude that $j \geq n - j_{i+1}^b$.

In the online S , any job scheduled "below" $k_1 + k_2 + \dots + k_i$ will receive at least $(1/\alpha k_i) \geq (1/\alpha b)$ bandwidth. Jobs scheduled between $k_1 + k_2 + \dots + k_i$ and $k_1 + k_2 + \dots + k_i + b$ receive bandwidth at least equal to $1/\alpha b$. So all jobs which the online schedules "below" $k_1 + k_2 + \dots + k_i + b$ receive at least $1/\alpha b$ bandwidth. This means r_{i+1}^b jobs receive at least $1/\alpha b$ bandwidth. Lemma 3.1 shows that $r_{i+1}^b \geq j_{i+1}^b$. The largest index for which $S_{(j)} < 1/\alpha b$ would be $n - r_{i+1}^b < j$. It follows that $S_{(j)} \geq 1/\alpha b$, and $\alpha S_{(j)} \geq T'_{(j)}$. ■

Coordinate-wise competitiveness immediately implies supermajorization, so we know that $S \prec^\alpha T'$. We still need to show that our bandwidth assignment was legal; in other words, that we did not exceed one unit of bandwidth on any machine.

Lemma 3.3 *The total bandwidth on any machine is at most 1.*

Proof: For $i \geq 1$, the total bandwidth assigned to jobs “between” job $k_1 + k_2 + \dots + k_i$ and job $k_1 + k_2 + \dots + k_{i+1}$ is at most

$$k_i \frac{1}{\alpha k_i} + \frac{1}{\alpha(k_i + 1)} + \frac{1}{\alpha(k_i + 2)} + \dots + \frac{1}{\alpha(k_{i+1})}$$

which is bounded by

$$\frac{1 + (\log k_{i+1}) - (\log k_i)}{\alpha}$$

The bandwidth assigned to the first through k_1 st jobs on the machine is bounded by $(1/\alpha)(1 + (1/2) + (1/3) + \dots + 1/k_1)$, or $(\log k_1)/\alpha$.

The sum over all values of i from 0 to $1 + \log m$ telescopes to

$$\frac{\log m + \log k_{1+\log m}}{\alpha} = \frac{\log m + \log t}{\alpha} = 1$$

Lemma 3.1 shows that $r_{1+\log n}^0$ encompasses all the jobs, so the overall total bandwidth on a machine is at most 1 as desired. ■

We now proceed to prove our desired result; the greedy algorithm is globally $O(\log n)$ -fair.

Theorem 3.3 *Let S^l represent the assignment of jobs to machines and bandwidths to jobs obtained by running the greedy online algorithm and then splitting bandwidth equally on each machine. For any other possible assignment T (including the globally fair assignment), $S^l \prec^\alpha T$ where $\alpha = \log m + \log t$.*

Proof: Let S represent the solution which assigns bandwidth by the method described above. Let T' represent the assignment T with bandwidths equalized. Lemma 3.2 guarantees that $S \prec^\alpha T'$. Lemma 3.3 guarantees that we have a legal bandwidth assignment. We now apply theorem 3.2 to show that $S^l \prec^1 S$ and $T' \prec^1 T$. Two applications of theorem 2.2 suffice to prove $S^l \prec^\alpha T$ as desired. ■

Finally, we notice that $t \leq n$. We can assume $m \leq n$ since having machines upon which solution T places no jobs can only help our algorithm. It follows that $\alpha \leq 2 \log n$. Thus we have proven global $O(\log n)$ -fairness for the greedy algorithm.

The following theorem follows from remark 3.1.

Theorem 3.4 *The greedy online algorithm results in a schedule that is globally $O(\log m)$ -balanced.*

Lower bounds Any online algorithm must be globally $\Omega(\log m)$ -fair as well as globally $\Omega(\log m)$ -balanced.

4 Open Problems

In the job centric model, it would be interesting to either prove global $O(\log m)$ -fairness or prove a lower bound of $\Omega(\log n)$, where m is the number of machines and n the number of jobs.

Acknowledgement

The first author would like to thank David Tse for introducing him to the concept of majorization.

References

- [1] Y. Afek, Y. Mansour, and Z. Ostfeld. Convergence complexity of optimistic rate based flow control algorithms. *Proceedings of the 28th ACM Symposium on Theory of Computing*, pages 89–98, 1996.
- [2] Y. Afek, Y. Mansour, and Z. Ostfeld. Phantom: A simple and effective flow control scheme. *ACM SIGCOMM*, pages 169–182, 1996.
- [3] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, and O. Waarts. On-line load balancing with applications to machine scheduling and virtual circuit routing. *25th ACM Symposium on Theory of Computing*, pages 623–31, 1993.
- [4] B. Awerbuch, Y. Azar, and S. Plotkin. Throughput competitive online routing. *34th IEEE symposium on Foundations of Computer Science*, pages 32–40, 1993.
- [5] B. Awerbuch and Y. Shavitt. Converging to approximated max-min flow fairness in logarithmic time. *Proceedings of the 17th IEEE Infocom conference*, pages 1350–57, 1998.
- [6] Y. Azar, J. Naor, and R. Rom. The competitiveness of on-line assignment. *Proceedings of the 3rd ACM-SIAM Symposium on Discrete Algorithms*, pages 203–210, 1992.
- [7] A. Goel, A. Meyerson, and S. Plotkin. Combining fairness with throughput: Online routing with multiple objectives. *32nd ACM Symposium on Theory of Computing*, 2000.
- [8] R. Graham. Bounds for certain multiprocessing anomalies. *Bell System Tech. J.*, 45:1563–81, 1966.
- [9] G.H. Hardy, J.E. Littlewood, and G. Polya. Some simple inequalities satisfied by convex functions. *Messenger Math.*, 58:145–152, 1929.
- [10] G.H. Hardy, J.e. Littlewood, and G. Polya. *Inequalities*. 1st ed., 2nd ed. Cambridge University Press, London and New York., 1934, 1952.
- [11] J. Kleinberg, Y. Rabani, and E. Tardos. Fairness in routing and load balancing. *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, 1999.
- [12] M.O. Lorenz. Methods of measuring concentrations of wealth. *J. Amer. Statist. Assoc.*, 9:209–219, 1905.
- [13] A.W. Marshall and I. Olkin. *Inequalities: theory of majorization and its applications*. Academic Press (Volume 143 of Mathematics in Science and Engineering), 1979.