An Improved Lower Bound for Load Balancing of Tasks with Unknown Duration

Yuan Ma [1] Serge Plotkin [2]

Abstract

Suppose there are n servers and a sequence of tasks, each of which arrives in an on-line fashion and can be handled by a subset of the servers. The level of the service required by a task is known upon arrival, but the duration of the service is unknown. The on-line load balancing problem introduced by Azar, Broder, and Karlin is to assign each task to an appropriate server so that the maximum load on the servers is minimized. Azar, Broder, and Karlin proved a lower bound of $\Omega(n^{1/2})$ on the competitive ratio for this problem. However, their lower bound argument used a sequence of tasks with exponential duration, and therefore this did not preclude a possible competitive ratio of the form $\text{poly}(\log T)$ where T denotes an upper bound on the duration of each task. In this paper, we prove a lower bound of $\Omega(\min\{n^{1/4}, T^{1/2}\})$, thereby proving that a $\text{poly}(\log T)$ competitive ratio is not possible. This should be compared to the analogous case for known-duration tasks, where it is possible to achieve $O(\log nT)$-competitive ratio.

1 Introduction

The following load-balancing problem was studied by Azar, Broder, and Karlin [2]. Suppose there are n servers and a sequence of tasks, each of which arrives in an on-line fashion and can be handled by a subset of the servers. The level of the service required by a task is known upon arrival, but the duration of the service is unknown. The goal is to assign each task to an appropriate server so that the maximum load on the servers is minimized. Azar, Broder, and Karlin proved a lower bound of $\Omega(n^{1/2})$ on the *competitive ratio* of any (deterministic or randomized) on-line algorithm, which in this case is defined as the ratio between the maximum load of the on-line algorithm and the maximum load of the optimal off-line algorithm [1]. They also described an $O(n^{2/3})$-competitive algorithm for this problem.

The upper bound was improved by Azar, Kalyanasundaram, Plotkin, Pruhs, Waarts [3], who showed a tight $O(n^{1/2})$ bound. This bound should be compared to the (much better) $O(\log n)$ bound of [1] for the case where the tasks have infinite durations (the "permanent tasks" case), and $O(\log nT)$ of [3] for the case where the duration of each task becomes known upon its arrival (the "known duration" case), where T denotes an upper bound on the duration of each task.

The lower bound of [2] used a sequence of tasks that have durations exponential in n, and thus it does not preclude a possible upper bound of the form $\text{poly}(\log T)$, similar to the bound for the known duration case. Since in practice T tends to be rather small, especially when compared to exponential in

[1] Department of Computer Science, Stanford University, Stanford, CA 94305. Email: yuan@cs.stanford.edu. Supported by an NSF Mathematical Sciences Postdoctoral Research Fellowship.

[2] Department of Computer Science, Stanford University, Stanford, CA 94305. Email: plotkin@cs.stanford.edu. Supported by ARO Grant DAAH04-95-1-0121, NSF Grant CCR-9304971, and Terman Fellowship.

[1] The competitive ratio concept was introduced in [8] and further developed in [5, 4, 6].

n, whether an algorithm can achieve a competitive ratio of sub-polynomial in T remains an interesting open question.

In this paper, we prove a lower bound of $\Omega(\min\{n^{1/2}, T^{1/2}\})$, thereby proving that a competitive ratio sub-polynomial in T is not possible. In the context of competitive circuit routing in ATM networks, our lower bound can be adapted to an $\Omega(\min\{n^{1/4}, T^{1/2}\})$ lower bound on the competitive ratio. (Note the term $n^{1/4}$, as opposed to $n^{1/2}$, and see [7] for more information on the on-line routing problem.) Like the lower bound in [2], our new lower bound holds even if each of the involved tasks can only require a service level 1 and if the algorithm is randomized. Unlike the argument in [2], our new proof is quite intuitive and much simpler.

2 Lower Bound

In this section, we first prove the main result for a deterministic algorithm. Then we extend it to a randomized algorithm.

Theorem 1 *The competitive ratio of any deterministic on-line algorithm is at least* $\Omega(\min(n^{1/2}, T^{1/3}))$.

Proof Let $r = \lfloor \min\{n^{1/2}/2, T^{1/3}\} \rfloor$. We show that an arbitrary deterministic on-line algorithm $\mathcal{A}$ must have a competitive ratio at least r. Consider an adversary who uses the following rules to choose a sequence of r^3 tasks each requiring a service of level 1:

1. assume a discrete time series and exactly one task arrives at the beginning of each time step;
2. the sequence is presented in r^2 phases each of which consists of r tasks;
3. at most one task in a phase stays until time r^3 and all other tasks in a phase depart immediately at the end of the phase;
4. within phase i $(1 \le i \le r^2)$ the jth task $(1 \le j \le r)$ needs to be served by either server $r+i$ or server j;
5. the only task (if there is any) of phase i $(1 \le i \le r^2)$ that is to stay until time r^3 is the first task (if there is any) in the phase that is scheduled by algorithm $\mathcal{A}$ to any of the first r servers.

First note that such a sequence is legitimate. That is, the adversary never uses a server whose label is larger than $r + r^2$ (which is at most n) or a task with duration longer than r^3 (which is at most T). Second, due to rules 3–4, an optimal algorithm can maintain a service level at most 1 on all the servers by letting server $r+i$ (for all i such that $1 \le i \le r^2$) to serve only the unique task (if any) of phase i that will stay until time r^3. On the other hand, since each task in phase i can be served by either server $r+i$ or one of the first r servers, if algorithm $\mathcal{A}$ is to avoid a competitive ratio at least r, during phase i $\mathcal{A}$ cannot assign all the r tasks to server $r+i$ and has to schedule at least one task to one of the first r servers. However, since there are r^2 phases, the first r servers will have at least r^2 tasks at the end. Thus, one of the first r servers must have at least $r^2/r = r$ tasks, implying a competitive ratio of at least r. ■

We next extend the lower bound to a randomized algorithm. To obtain the strongest possible result, we consider an *oblivious* adversary, who has no access to the coin-tosses of a randomized algorithm.

Theorem 2 *The competitive ratio of any randomized on-line algorithm against an oblivious adversary is at least* $\Omega(\min(n^{1/2}, T^{1/3}))$.

Proof We will modify the adversary for Theorem 1 so that it will fool a randomized algorithm $\mathcal{A}$ with high probability. Let r be defined as in the proof of Theorem 1, and we will show that $\mathcal{A}$ has a competitive ratio at least $r/2$. Our new adversary will follow all the rules described in the proof of Theorem 1 except rule 5, which cannot be followed explicitly as it is since the new adversary should be oblivious and thus has no access to the random coin tosses of algorithm $\mathcal{A}$. Now the adversary is left with a set of sequences, say S, as the candidates. (The freedom comes from the fact that the adversary has yet to decide which task will stay until time r^3.) Since all sequences in S look the same to $\mathcal{A}$ up to the end of time step r, $\mathcal{A}$ must behave the same in phase 1 on all sequences in S. (Of course, the real behavior of $\mathcal{A}$ depends on its random coin tosses, but the coin-tossing method should be independent of the particular choice of sequence in S.) Now, at least one task in phase 1, say task i_1, should be scheduled by $\mathcal{A}$ to one of the first r servers with probability at least 1/2, since otherwise the expected number of tasks on server $r+1$ would be at least $r/2$ and a desired lower bound would follow immediately. (Like argued in the proof of Theorem 1, the optimal off-line algorithm can easily maintain a service level at most 1 on any sequence in S.) Let S_1 be the subset of S consisting of all sequences whose i_1th task in phase 1 will stay until time r^3. Next consider S_1 (instead of S) and phase 2 (instead of phase 1) to find a task i_2 of phase 2 that should be scheduled by $\mathcal{A}$ to one the first r servers with probability at least 1/2. Then define S_2 to be the subset of S_1 consisting of all sequences whose i_2th task in phase 2 will be the one to stay until time r^3. We proceed in this fashion until we have constructed S_{r^2}, which will consist of a single sequence s. On sequence s, the expected number of tasks scheduled by $\mathcal{A}$ to one of the first r servers is at least $r^2/2$. Thus, the competitive ratio of $\mathcal{A}$ is at least $(r^2/2)/r = r/2$, as claimed. ■

References

[1] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, and O. Waarts. On-line machine scheduling with applications to load balancing and virtual circuit routing. In *Proc. 25th Annual ACM Symposium on Theory of Computing*, pages 623–631, May 1993.

[2] Y. Azar, A. Broder, and A. Karlin. On-line load balancing. In *Proceedings of the 33rd Annual IEEE Symposium on Foundations of Computer Science*, pages 218–225, October 1992.

[3] Y. Azar, B. Kalyanasundaram, S. Plotkin, K. Pruhs, and O. Waarts. On-line load balancing of temporary tasks. In *Proceedings of Workshop on Algorithms and Data Structures*, pages 119–130, August 1993.

[4] A. Borodin, N. Linial, and M. Saks. An optimal online algorithm for metrical task systems. *J. ACM*, (39):745–763, 1992.

[5] A.R. Karlin, M.S. Manasse, L.Rudolph, and D.D. Sleator. Competitive snoopy caching. *Algorithmica*, 1(3):70–119, 1988.

[6] M.S. Manasse, L.A. McGeoch, and D.D. Sleator. Competitive algorithms for online problems. In *Proc. 20th Annual ACM Symposium on Theory of Computing*, pages 322–332, 1988.

[7] S. Plotkin. Competitive routing in ATM networks. *IEEE J. Selected Areas in Comm.*, pages 1128–1136, August 1995. Special issue on Advances in the Fundamentals of Networking.

[8] D.D. Sleator and R.E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.