Journal of Computer and System Sciences 91 (2018) 42–68

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Rejecting jobs to minimize load and maximum flow-time ☆

Anamitra Roy Choudhury [a,*], Syamantak Das [b,1], Naveen Garg [b], Amit Kumar [b]

[a] IBM Research-India, New Delhi 110070, India
[b] Department of Computer Science and Engineering, IIT Delhi, New Delhi 110016, India

ARTICLE INFO

Article history:
Received 2 April 2016
Accepted 13 July 2017
Available online 1 September 2017

Keywords:
Online job scheduling
Restricted assignment
Flow-time
Rejection model
Competitive ratio

ABSTRACT

The notion of competitive ratio often turns out to be too pessimistic for the analysis of online algorithms. Although the approach of resource augmentation (introduced by Kalyanasundaram and Pruhs) has been very successful in dealing with a variety of objective functions, there are problems for which even a (arbitrary) constant speedup cannot lead to a constant competitive algorithm. Here we propose a rejection model which permits the online algorithm to not serve epsilon-fraction of requests. We present $O(\log^2 1/\varepsilon)$ and $O(1/\varepsilon^4)$-competitive algorithms for the problems of load balancing and minimizing maximum flow time in the restricted assignment setting.

1. Introduction

Online algorithms are usually analyzed using the notion of competitive ratio which compares the solution obtained by the algorithm to that obtained by an offline adversary for the worst possible input sequence. Researchers have tried to address the criticism of this measure being too pessimistic by either limiting the power of the adversary – oblivious adversary, stochastic adversary – or giving more power to the online algorithm – lookahead, additional resources, etc. One popular approach especially for scheduling problems has been that of "resource augmentation" and was first proposed by Kalyanasundaram and Pruhs [29]. In this model the machines of the online algorithm have more speed than those of the offline algorithm. Many scheduling problems for which no constant competitive online algorithm is possible now have such algorithms in this resource augmentation model. The success of the speed augmentation model lies in the fact that many natural algorithms can be analyzed in this framework.

In this paper, we propose a "rejection model" in which there is no resource augmentation, but we allow the online algorithm to not serve an ε-fraction of requests. There are two principal reasons for considering this model: (i) although resource augmentation has been very successful in dealing with a variety of objective functions, there are problems for which even a (arbitrary) constant speedup cannot lead to a constant competitive algorithm – we consider two such problems in this paper, and (ii) this might be a natural assumption in many settings where job rejection is part of the service provided by a system (e.g., "Server busy: Please try again later" message we often see when accessing popular websites).

☆ A preliminary version of this paper appeared in the Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA.

* Correspondence to: IBM Research-India, Plot-4, Phase-II, Block-C, Institutional Area, Vasant Kunj, New Delhi 110070, India.
E-mail addresses: anamchou@in.ibm.com (A.R. Choudhury), syamanta@uni-bremen.de (S. Das), naveen@cse.iitd.ac.in (N. Garg), amitk@cse.iitd.ac.in (A. Kumar).
[1] Present Address: University of Bremen, Bremen 28359, Germany.

http://dx.doi.org/10.1016/j.jcss.2017.07.006
0022-0000/

For most scheduling problems, an algorithm in the resource augmentation model can be "simulated" in this rejection model by roughly letting each machine drop every $(1/\varepsilon)$th job assigned to it. However, the rejection model is much more powerful than the resource augmentation model since we are not restricted to drop an ε-fraction of jobs assigned to each machine, i.e., we could drop many more jobs assigned to one machine as compared to another as long as the overall number of rejected jobs stays within ε-fraction of the total number of jobs. We demonstrate this by considering two classical problems – load balancing and maximum (weighted) flow time. For both these problems we give constant competitive algorithms in the rejection model while no such algorithms are possible in the resource augmentation model. This is the key contribution of this paper.

The problems considered in this paper are in the restricted assignment setting where each job can be assigned only to a subset of machines. All our algorithms are pre-emptive, immediate dispatch – a job is assigned to a machine as soon as it is released and non-migratory – a job is processed only on the machine to which it is assigned. While ideally we would also like to make rejection decisions immediately when the job is released, we also allow for the job being rejected while it is waiting in the queue. We justify this by showing that no online algorithm with immediate dispatch and immediate rejection can be constant competitive for the load balancing problem (and hence, the maximum flow-time problem).

Our results For the load balancing problem (`LoadBalancing`) where the objective is to minimize the maximum load on any machine, we give an $O(\log\left(\frac{1}{\varepsilon}\right))$ competitive immediate rejection algorithm when all jobs have unit processing time and the online algorithm can reject an ε-fraction of the jobs. For general processing times our algorithm is not immediate reject and is $O(\log^2\left(\frac{1}{\varepsilon}\right))$-competitive. We show that one cannot get a constant competitive algorithm if we require immediate rejection. Note that there is a $\Omega(\log m)$ lower bound on the competitive ratio of any online algorithm for this problem where m is the number of machines [9]. Further, making the machines an ε-fraction faster has no significant impact on this lower bound.

For the maximum flow-time problem Anand et al. [5] show that no immediate dispatch algorithm can be constant competitive even when the jobs are unit length and we allow resource augmentation. For this setting of unit jobs we show an immediate reject algorithm which is $O(1/\varepsilon)$-competitive and rejects at most an ε fraction of the jobs. When jobs have weights (`WtdMaxFlowTime`), the objective is to minimize the maximum weighted flow-time of a job. For this setting Anand et al. [5] show that no online algorithm can be constant competitive even when we allow non-immediate dispatch and resource augmentation. Our algorithm for this setting is immediate dispatch but not immediate reject and is allowed to reject jobs of total weight at most an ε-times the total weight of all jobs and has a competitive ratio of $O(1/\varepsilon^4)$. We also show that it is not possible to get better than $O(1/\varepsilon)$-competitive algorithm in this model, and that one cannot get a good competitive algorithm if we are required to perform immediate reject and immediate dispatch.

We further generalize our result to a setting where each job has two kinds of weight – rejection weight and flow-time weight. The weighted flow-time of the job is defined as its flow-time times its flow-time weight, and the goal, as before, is to minimize the maximum weighted flow-time of a job. However, the total rejection weight of the jobs which get rejected should be at most ε fraction of the total rejection weight of all the jobs. We obtain an $O(1/\varepsilon^6)$-competitive algorithm for this problem. The problem of minimizing maximum stretch is a special case of this setting in the rejection model. Here, the rejection weights are all unit while the flow-time weights are the inverse of processing sizes.

2. Related work

Load balancing Graham [26] considered this problem in the context of identical machines (where each job can be assigned to any machine: processing time of a job on any machine is same) and showed that the simple greedy heuristic of assigning the next task to the least loaded machine is 2-competitive (see also the survey by Azar [8]). Albers [1] improved the competitive ratio to 1.923 and also showed a lower bound of 1.852 on the competitive ratio of any deterministic online algorithm, while Albers et al. [2] shows improved bounds in the special case where the online algorithm knows the sum of job sizes at any point. For related machines model (every machine in this model has a speed, and a job of size p requires a processing time of p/s when assigned to a machine of speed s), Berman et al. [14] gave constant competitive algorithms. However, the problem becomes significantly harder in the unrelated machines model, where a job can have different processing time on different machines. Azar et al. [9] considered the problem in the restricted assignment setting, and gave an $O(\log m)$-competitive algorithm for load balancing (m being the number of machines). They also complemented this result by proving lower bound of $\Omega(\log m)$ for any deterministic and $\Omega(\ln m)$ for any randomized online algorithm under the restricted assignment model. Buchbinder et al. [15] gave an alternative, more general upper bound on the load on any prefix of the most loaded machines. For the unrelated machines setting, Aspnes et al. [6] gave an $O(\log m)$-competitive algorithm. There has been some work on resource augmentation in this setting. Azar et al. [10] showed a competitive ratio of $1 + 1/2^{\frac{n}{m}(1-o(1))}$ when the online algorithm is allowed to use n identical machines while the offline optimal is restricted to $m < n$ identical machines.

Flow-time minimization There has been considerable work on scheduling with the objective of minimizing a suitable norm of the flow-time of jobs. For the objective of average flow-time of jobs, a logarithmic competitive algorithm in the identical machines setting is known [30,7]. Garg and Kumar [23] extended this result to the related machines setting. Garg and