Online Scheduling with Hard Deadlines on Parallel Machines

Jihuan Ding\textsuperscript{1,2} and Guochuan Zhang\textsuperscript{1}

\textsuperscript{1} Department of Mathematics, Zhejiang University
Hangzhou 310027, China
\{dingjihuan, zgc\}@zju.edu.cn
\textsuperscript{2} College of Operations Research and Management Science,
Qufu Normal University, Rizhao 276826, China

\textbf{Abstract.} In this paper, motivated by on-line admission control in the hard deadline model, we deal with the following scheduling problem. We are given $m$ identical machines (multi-streams). All jobs (requests) have identical processing time. Each job is associated with a release time and a deadline, neither of which is known until the job arrives. As soon as a job is available, we must immediately decide if the job is accepted or rejected. If a job is accepted, then it must be completed no later than its deadline. The goal is to maximize the total number of jobs accepted. The one-machine case has been extensively studied while little is known for multiple machines. Our main result is deriving a nontrivial optimal online algorithm with competitive ratio $\frac{3}{2}$ for the two-machine case by carefully investigating various strategies. Deterministic lower bounds for the general case are also given.

1 Introduction

\textbf{Problem statement.} We deal with the following real-time scheduling problem with online admission control. We are given $m$ identical machines. Each job $j$ arrives at its release time $r_j$ that is not known in advance. Upon arrival of job $j$ its deadline $d_j$ is revealed. All jobs have equal processing times of 1. Preemption is not allowed. At any time when some machine is idle, we have to decide whether to start an “accepted” job or not, and if so, to choose which one, based only on the information on the jobs released so far. Those jobs that can not be scheduled to meet their deadlines will be lost (not processed at all). The objective is to maximize the number of completed jobs, i.e., the number of jobs meeting their deadlines. We also call this objective throughput maximization.

To evaluate online algorithms we adopt the standard measure of \textit{competitiveness}. An online algorithm is $c$-\textit{competitive} if on every input instance the number of early jobs by the algorithm is at least $\frac{1}{c}$ times that of an optimum schedule.

\textbf{Previous work.} The throughput maximization problem on a single machine has been extensively studied in the literature. Goldman \textit{et al.} \cite{6} gave a lower
bound of $\frac{3}{2}$ on the competitive ratio of randomized algorithms and the tight bound of 2 for deterministic algorithms for the online problem. They further proved that a greedy algorithm is a $\frac{3}{2}$-competitive if $d_j - r_j \geq 2$ for all jobs $j$, which implies that the lower bound of 2 can be beaten if the jobs have sufficiently large slack. Along this line, Goldwasser [3] made a parameterized extension of this result: if $d_j - r_j \geq \lambda$ for all jobs $j$, where $\lambda > 0$ is a real number, then the competitive ratio is $1 + \frac{1}{\lambda}$. In 2003 Goldwasser and Kerbikov [4] extended the previous results [3] under a reasonable assumption called *immediate notification*. This assumption requires the scheduler to determine whether to accept job $j$ immediately at its arrival. If the job is accepted it must be completed by this deadline.

Chrobak et al. [2] considered randomization and restarts for online scheduling of equal-length jobs. They gave a $\frac{5}{3}$-competitive randomized algorithm. For the restart model (allowed to abort a job during execution and an aborted job can be restarted and completed later), they presented an optimal $\frac{3}{2}$-competitive algorithm.

In contrast, to our best known, there are few results for the throughput maximization problem on parallel machines. The offline problem on $m$ machines can be solved in polynomial time [1]. For the online case Lee [7] considered the problem of maximizing the sum of the length of accepted jobs on $m$ machines, where each job of length $L$ can be delayed for at least $k \cdot L$ for $k < 1$ before it is started, while still meeting its deadline. He presented an $O(\log(1/k))$-competitive randomized algorithm for $m$ machines where $m = 1, 2, \ldots, O(\log(1/k))$. For $m \geq 2$ a $[m + 1 + m \cdot (1/k)^{(1/m)}]$-competitive deterministic algorithm was derived.

Our results. We first derive simple deterministic lower bounds for $m$ parallel machines. Then we show that the competitive ratio of a simple algorithm is exactly two. Our main result is an optimal non-trivial on-line algorithm with competitive ratio of $3/2$ for the two-machine case (very recently Goldwasser and Pedigo [5] obtained the same ratio independently). Moreover all our results are still valid under the assumption of *immediate notification*.

## 2 Lower Bounds for $m \geq 2$ Machines

**Theorem 1.** For on-line scheduling of unit-length jobs on $m$ machines to maximize the number of jobs completed, the competitive ratio of any deterministic online algorithm is not smaller than

$$R = \begin{cases} 
3/2 & \text{if } m = 2 \\
6/5 + 1/(5k) & \text{if } m = 3k \\
(6k + 2)/(5k + 1) & \text{if } m = 3k + 1, k \geq 1. \\
(6k + 3)/(5k + 2) & \text{if } m = 3k + 2
\end{cases}$$

**Proof.** We only prove the lower bound for $m = 2$. At time 0, a job with deadline 5 comes. For any algorithm $A$, let $t(0 \leq t \leq 4)$ denote the start time of the job (If the algorithm does not start the job by time 4, the job will not be scheduled. It