Online Algorithm for Parallel Job Scheduling and Strip Packing*

Johann L. Hurink and Jacob Jan Paulus

University of Twente, P.O. box 217, 7500AE Enschede, The Netherlands,
Fax: +31534894858
j.j.paulus@utwente.nl

Abstract. We consider the online scheduling problem of parallel jobs on parallel machines, $P|\text{online}-\text{list}, m_j|C_{\max}$. For this problem we present a 6.6623-competitive algorithm. This improves the best known 7- competitive algorithm for this problem. The presented algorithm also applies to the special case where machines are ordered on a line and only adjacent machines can be assigned to a job and, therefore, also to online orthogonal strip packing. Since previous results for online orthogonal strip packing assume bounded rectangles, the presented algorithm is the first with a constant competitive ratio.

1 Introduction

Consider the following online machine scheduling problem. Jobs $j = 1, 2, ..., n$ are presented one by one to the decision maker and are characterized by their processing time and the number of machines simultaneously required for processing. Job j has processing time p_j and requires simultaneously m_j out of the available m machines. As soon as a job becomes known, it has to be scheduled irrevocably (i.e. its start time has to be set) without knowledge of successive jobs. Preemption is not allowed and the objective is to minimize the makespan.

Using the three-field notation introduced in [2], this problem is denoted by $P|\text{online}-\text{list}, m_j|C_{\max}$, see also [5,6]. Note that sometimes $size_j$ is used instead of m_j to denote the parallel machine requirement of job j.

The quality of an online algorithm is measured by its competitive ratio. An online algorithm is called ρ-competitive if for any sequence of jobs it produces a schedule with makespan at most ρ times the makespan of the optimal schedule. For background on online scheduling see [6].

The problem $P|\text{online}-\text{list}, m_j|C_{\max}$ gained considerable attention in the last few years. It was pointed out by Johannes [5] that a greedy algorithm which schedules the jobs as early as possible, has a competitive ratio of m. She was also the first to design an online algorithm with a constant competitive ratio, which has a competitive ratio of 12. This result was successively improved by Ye and Zhang, first to an 8 and later to a 7-competitive algorithm [7,8]. For the special case with only 2 machines an greedy algorithm is optimal [4], i.e.

* Part of this research has been funded by the Dutch BSIK/BRICKS project.

C. Kaklamanis and M. Skutella (Eds.): WAOA 2007, LNCS 4927, pp. 67–74, 2008.

no online algorithm for $P2|\text{online}-\text{list}, m_j|C_{\max}$ with competitive ratio strictly less than 2 exists.

Far less is known about lower bounds for the general m machine case. In [4] an ILP formulation is presented to derive lower bounds, and by means of an ILP solver a lower bound of 2.43 is derived. The best analytical lower bound is the bound of 2 from the two machine case.

In the literature also semi-online cases have been studied, e.g. *jobs appear with non-increasing processing times*, *jobs appear with non-increasing machine requirement* or the *largest processing time is known*. For these semi-online problems the gap between the lower and the upper bound on the competitive ratio is much smaller, see [7,8]. Variations on the scheduling model, where jobs are *malleable* or *preemption* is allowed, or with different online paradigms such as *non-clairvoyance* and *online-time*, are also considered in the literature. For an overview of these various models see [5,6,8].

The problem $P|\text{online}-\text{list}, m_j|C_{\max}$ resembles online orthogonal strip packing. The difference lies in the following. In the scheduling problem any choice of m_j machines for processing job j is allowed, where in strip packing rectangles cannot be split. If the machines were to be ordered on a line and job j requires m_j *adjacent* machines for its processing, the problems become the same. As it turns out, the analysis of the online algorithm presented in this paper also hold in the presence of such a machine ordering and adjacency requirement. Therefore, the presented online algorithm applies to online orthogonal strip packing as well. Till now, the performance ratio of the best online algorithm for online orthogonal strip packing is 6.99, which is due to Baker and Schwarz [1]. It is worthwhile to mention, that the existing bounds for orthogonal strip packing are attained under the assumption that the rectangles have height at most 1. Analogous, the processing time of the jobs is bounded by 1. To the best of our knowledge, the presented algorithm is the first online algorithm with constant competitive ratio for orthogonal strip packing without knowledge of the overall maximum processing time of a job.

The presented approach in this paper leads to a new online algorithm for $P|\text{online}-\text{list}, m_j|C_{\max}$. The algorithm takes two parameters, one parameter defines the borderline between *big* jobs (jobs with large m_j) and *small* jobs, and the second parameter defines *classes of processing times*. Small jobs with processing times of the same class get scheduled in parallel. A proper choice of the two parameters leads to an online algorithm that has a competitive ratio of at most $\frac{7}{2}+\sqrt{10}(\approx 6.6623)$.

In Section 2, we present the online algorithm and prove that it has a competitive ratio of at most 6.6623. In Section 3, we show that the algorithm also applies to the online orthogonal strip packing problem.

2 The Online Algorithm

Before we present the algorithm and the proof of its competitive ratio, we introduce some notation and basic results.

Given a sequence of jobs $\sigma = (1, 2, ..., n)$ we can derive two lower bounds on the makespan of the optimal offline schedule, denoted by $OPT(\sigma)$. On the one hand, the optimal makespan is bounded by the length of the longest job in σ, i.e. $OPT(\sigma) \geq \max_{j=1}^{n}\{p_j\}$. On the other hand, if the work load of a job j is given by $m_j \cdot p_j$, then the total work load divided by m is a lower bound on $OPT(\sigma)$, i.e. $OPT(\sigma) \geq \frac{1}{m}\sum_{j=1}^{n} m_j \cdot p_j$.

Let $\mathcal{S}(\sigma)$ be the schedule created by an online algorithm and denote its makespan by $ON(\sigma)$. For a collection of disjoint intervals X from $[0, ON(\sigma)]$, we denote by $|X|$ the cumulative length of the intervals in X.

The next lemma follows directly from the above presented lower bounds on $OPT(\sigma)$.

Lemma 1. *If* $[0, ON(\sigma)]$ *can be partitioned in* X *and* Y *such that* $|X| \leq x \cdot \max_{j=1}^{n}\{p_j\}$ *and* $|Y| \leq y \cdot \frac{1}{m}\sum_{j=1}^{n} m_j \cdot p_j$, *then* $ON(\sigma) \leq (x + y) \cdot OPT(\sigma)$.

In the following, we design an online algorithm for $P|\text{online} - \text{list}, m_j|C_{\max}$ such that the constructed schedules can be partitioned in X and Y as in Lemma 1 such that $x+y$ is small. To do this, we distinguish between two types of jobs; jobs with a large machine requirement and jobs that require only a few machines for processing. A job j is called *big* if it has machine requirement $m_j \geq \lceil \alpha \cdot m \rceil$ with $\alpha \in (0, \frac{1}{2}]$, and called *small* otherwise. This is a generalization of the distinction between big and small jobs found in [5,7,8], where α is a priori fixed to either $\frac{1}{2}$ or $\frac{1}{3}$. Furthermore, the small jobs are classified according to their length. A small job j belongs to job class J_k if $\beta^k \leq p_j < \beta^{k+1}$, where $\beta > 1$ is the second parameter of the algorithm. Note that k may be negative. Similar classifications can be found in *Shelf Algorithms* for Strip Packing [1], which are applied to group rectangles of similar height. The online algorithm to be described in the following, takes α and β as parameters and is denoted by $ON_{\alpha,\beta}$.

In the schedules created by the online algorithm $ON_{\alpha,\beta}$, big jobs are never scheduled in parallel to other jobs, and (where possible) small jobs are put in parallel to other small jobs of the same job class. The intuition behind the online algorithm $ON_{\alpha,\beta}$ is the following. Big jobs have a relative high average load and small jobs are either grouped together to a high average load or there is a small job with a relative long processing time. In the proof of Theorem 1, the intervals with many small jobs, together with the intervals with big jobs will be compared to the *work load bound* for $OPT(\sigma)$ (the Y part for Lemma 1), and the intervals with only a few small jobs are compared to the *longest job bound* for $OPT(\sigma)$ (the X part for Lemma 1).

The following gives a precise description of the algorithm $ON_{\alpha,\beta}$. The algorithm creates schedules with sparse intervals S_k and dense intervals D_k^i for the small jobs of class J_k. With n_k we count the number of dense intervals created for J_k. All small job scheduled in such an interval $[a, b)$ start at a. As a consequence, job j *fits* in interval $[a, b)$ if the machine requirement of the jobs already in $[a, b)$ plus m_j is at most m.

Algorithm $ON_{\alpha,\beta}$**:**
When scheduling job j and

1. job j is small, i.e. $m_j < \lceil \alpha \cdot m \rceil$, and belongs to job class J_k. Try in the given order:
   - Schedule job j in the first D_k^i where it fits.
   - Schedule job j in S_k.
   - Let $n_k := n_k + 1$ and S_k becomes $D_k^{n_k}$. Create a new interval S_k at the end of the current schedule with length β^{k+1}. Schedule job j in S_k.
2. job j is big, i.e. $m_j \geq \lceil \alpha \cdot m \rceil$. Schedule job j at the end of the current schedule.

The structure of the schedule created by $ON_{\alpha,\beta}$ is illustrated by Fig. 1. Note that at any time for each job class J_k there is at most one sparse interval S_k.

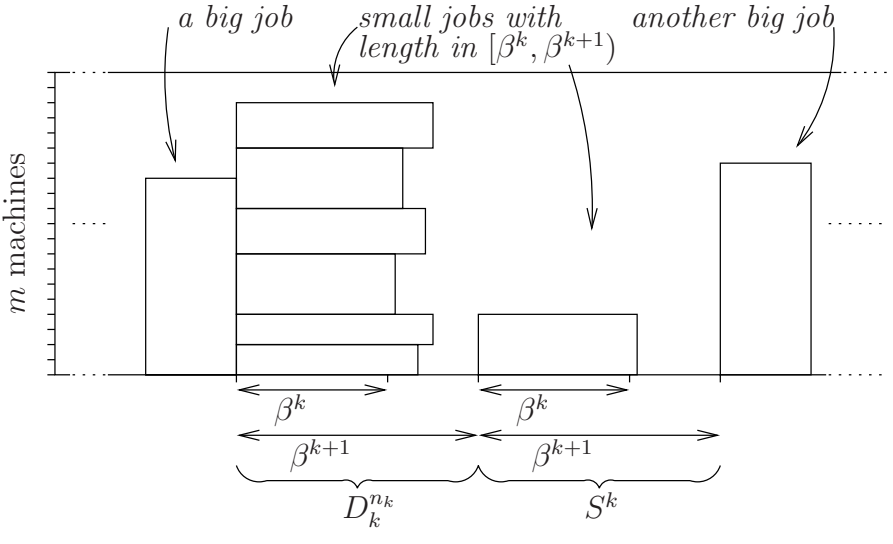

Fig. 1. Part of a schedule created by $ON_{\alpha,\beta}$

In the proof of Theorem 1 we will use the fact that the dense intervals D_k^i contain quite some load, i.e. there is a small job from job class J_k that did not fit in the dense intervals and had to be scheduled in a newly created sparse interval. When considering the length of the dense intervals, we take the load of both the dense and sparse intervals into account. Lemma 2 formalizes this. Slightly abusing notation, we will also refer to S_k and D_k^i as the set of jobs that are scheduled in intervals S_k and D_k^i.

Lemma 2. *For any* $\alpha \in (0, \frac{1}{2}]$ *and* $\beta > 1$*, the total work load in the dense and sparse intervals of schedule* $\mathbf{S}(\sigma)$ *created by* $ON_{\alpha,\beta}$*, is at least* $\frac{2m}{3\beta}$ *times the length of all dense intervals.*

Proof. Let σ be an arbitrary list of jobs and let $\mathbf{S}(\sigma)$ be the corresponding schedule constructed by the online algorithm $ON_{\alpha,\beta}$. Consider all dense intervals in $\mathbf{S}(\sigma)$ corresponding to one job class J_k. Since all jobs in D_k^i are scheduled to start at the beginning of interval D_k^i and have a processing time of at least β^k, the interval D_k^i has $\sum_{j\in D_k^i} m_j$ machines in use for at least the first $\frac{1}{\beta}$ fraction of D_k^i. If $\alpha \leq \frac{1}{3}$ this number of machines in use is larger than $\frac{2m}{3}$, and we are done. If $\alpha \in (\frac{1}{3}, \frac{1}{2}]$, we claim that for each job class J_k this number of machine in use is for at most one dense interval less than $\frac{2m}{3}$.

Let $\alpha \in (\frac{1}{3}, \frac{1}{2}]$ and let D_k^l be the first dense interval of job class J_k for which $\sum_{j\in D_k^i} m_j < \frac{2m}{3}$. After the creation of this dense interval, all newly created dense intervals for job class J_k, contain only small jobs with machine requirement $m_j > \frac{m}{3}$ (otherwise these jobs would have been scheduled in D_k^l or in an earlier dense interval). This implies that all successively created dense intervals for job class J_k have at least $\frac{2m}{3}$ machines in use. More precisely, they contain two small jobs with machine requirement $m_j > \frac{m}{3}$. Furthermore, the existence of D_k^l implies that S_k contains at least one job with machine requirement $m_j > \frac{m}{3}$.

So, for each job class J_k there is either one D_k^l with machine usage less than $\frac{2m}{3}$ and S_k contains a job with $m_j > \frac{m}{3}$, or all D_k^i have machine usage of at least $\frac{2m}{3}$. Thus, the total load of the small jobs in job class J_k is at least $\frac{2m}{3\beta}$ times the total length of all dense intervals corresponding to this job class. □

Next we will prove the upper bound on the performance guarantee of the online algorithm $ON_{\alpha,\beta}$.

Theorem 1. *For any $\alpha \in (0, \frac{1}{2}]$ and $\beta > 1$ the competitive ratio of the online algorithm $ON_{\alpha,\beta}$ for the problem $P|\text{online}-\text{list}, m_j|C_{\max}$ is at most $\max\{\frac{1}{\alpha}, \frac{3\beta}{2}\}+\frac{\beta^2}{\beta-1}$.*

Proof. Let σ be an arbitrary list of jobs and let $\mathbf{S}(\sigma)$ be the corresponding schedule constructed by the online algorithm $ON_{\alpha,\beta}$. We partition $[0, ON_{\alpha,\beta}(\sigma)]$ into three parts: The first part B consists of the intervals in which big jobs are scheduled, the second part D consists of the dense intervals, and finally the third part S contains the sparse intervals.

Since part B contains only jobs with machine requirement $m_j \geq \lceil \alpha \cdot m \rceil$, the total work load in B is at least $\alpha \cdot m \cdot |B|$. According to Lemma 2, the total work load in D and S is at least $\frac{2m}{3\beta} \cdot |D|$. This work load is also in the optimal offline schedule. Therefore, $\min\{\alpha \cdot m, \frac{2m}{3\beta}\} \cdot (|B| + |D|) \leq m \cdot OPT(\sigma)$, or equivalently

$$|B| + |D| \leq \max\{\frac{1}{\alpha}, \frac{3\beta}{2}\} \cdot OPT(\sigma) \ . \tag{1}$$

To simplify the arguments for bounding $|S|$, we normalize the jobs in $\mathbf{S}(\sigma)$ by letting J_0 be the smallest job class, i.e. the smallest processing time of the small jobs is between 1 and β. Then $|S_k| = \beta^{k+1}$. Let $\bar{k}$ be the largest k for which

there is a sparse interval in $\mathbf{S}(\sigma)$. Since there is at most one sparse interval for each job class J_k, the length of S is bounded by

$$|S| \leq \sum_{k=0}^{\bar{k}} |S_k| = \sum_{k=0}^{\bar{k}} \beta^{k+1} = \frac{\beta^{\bar{k}+2} - \beta}{\beta - 1} \ .$$

On the other hand, since $S_{\bar{k}}$ is not empty, we know that there is a job in $\mathcal{S}(\sigma)$ with processing time at least $\frac{|S_{\bar{k}}|}{\beta} = \beta^{\bar{k}}$. Thus,

$$|S| \leq \frac{\beta^2}{\beta - 1} \cdot OPT(\sigma) \ . \tag{2}$$

Using Lemma 1, (1) and (2) lead to the following bound on the makespan of the schedule created by online algorithm $ON_{\alpha,\beta}$:

$$ON_{\alpha,\beta}(\sigma) \leq \left(\max\{\frac{1}{\alpha}, \frac{3\beta}{2}\} + \frac{\beta^2}{\beta - 1} \right) \cdot OPT(\sigma) \ .$$

Thus, $ON_{\alpha,\beta}$ has a competitive ratio of at most $\max\{\frac{1}{\alpha}, \frac{3\beta}{2}\} + \frac{\beta^2}{\beta-1}$. □

To find the best possible performance bound of $ON_{\alpha,\beta}$, we have to find values of α and β which minimize the competitive ratio from Theorem 1.

Corollary 1. *The worst case bound for $ON_{\alpha,\beta}$ is minimal if $\alpha \geq \frac{10}{3(5+\sqrt{10})}$ (≈ 0.4084) and $\beta = 1 + \frac{\sqrt{10}}{5}$ (≈ 1.6325), leading to a competitive ratio of $\frac{7}{2} + \sqrt{10}$ (≈ 6.6623).*

Proof. If $\frac{1}{\alpha} > \frac{3\beta}{2}$ then by increasing the value of α, the value of $\max\{\frac{1}{\alpha}, \frac{3\beta}{2}\}$ can be decreased. Therefore, it is best to choose $\frac{1}{\alpha} \leq \frac{3\beta}{2}$. The competitive ratio then becomes $\frac{3\beta}{2} + \frac{\beta^2}{\beta-1}$. The optimal value for β can be found by differentiating this term. □

It is interesting to note that there is not just one setting of α and β that gives the best performance guarantee, but for $\beta = 1.6325$ all $\alpha \in [0.4084, 0.5]$ result in 6.6623-competitiveness of $ON_{\alpha,\beta}$.

3 Machines on a Line and Orthogonal Strip Packing

The presented online algorithm also applies to scheduling problems where the machines are ordered on a line and only adjacent machines can be assigned to a specific job. To let the presented algorithm apply to this case, we simply specify that whenever a job j is assigned to some interval, it is scheduled not only at the start of the interval, but also assigned to the *first* m_j machines available (first with respect to the line ordering of the machines). This way we can guarantee that each job j gets assigned to m_j adjacent machines and the algorithm still

gives the same schedule as before. To the best of our knowledge the presented online algorithm is the first with constant competitive ratio for this problem. For previous developed online algorithms for $P|\text{online} - \text{list}, m_j|C_{\max}$ no such adaptation makes them applicable to this special case.

Since the presented online algorithm also applies to this special case, it applies to the online orthogonal strip packing problem. The online orthogonal strip packing problem is a two-dimensional packing problem. Without rotation rectangles have to be packed on a strip with fixed width and unbounded height. The objective is to minimize the height of the strip used. In the online setting one rectangle is presented after the other and has to be assigned without knowledge of successive rectangles.

To see that these problems are equivalent, let the machines correspond to the width of the strip, and time to the height of the strip. The width of a rectangle j corresponds to the machine requirement of job j and its height to the processing time. Minimizing the height of the strip used is equivalent to minimizing the makespan of the machine scheduling problem.

Although most of the research on online orthogonal strip packing focuses on asymptotic performance ratios, Baker and Schwarz [1] developed a *Shelf Algorithm* that has competitive ratio 6.99 under the assumption that the height of a rectangle is at most 1. So, the presented algorithm not only improves the best known competitive ratio for online orthogonal strip packing, but also does not require the assumption on the bounded height.

4 Conclusions

In this paper we presented a new online algorithm for $P|\text{online} - \text{list}, m_j|C_{\max}$ with a competitive ratio of 6.6623. Due to the optimization of the parameters of $ON_{\alpha,\beta}$ a better online algorithm can only be found by employing new ideas, both in the design and analysis. There is room for improvement since the gap with the best lower bound (2.43) is large.

The presented algorithm also applies to the problem where the machines are ordered on a line and to online orthogonal strip packing. It is an interesting open question whether or not the additional requirement of a line ordering will lead to a different competitive ratio of the problem.

Note

In the independent work of Han et al. [3] the same results where obtained in the setting of online orthogonal strip packing.

References

1. Baker, B.S., Schwarz, J.S.: Shelf Algorithms for Two-Dimensional Packing Problems. SIAM Journal on Computing 12(3), 508–525 (1983)
2. Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G.: Optimization and approximation in deterministic sequencing and scheduling: A survey. Annals of Discrete Mathematics 5, 287–326 (1979)

3. Han, X., Ye, D., Zhang, G.: A note on online strip packing. Manuscript (2007)
4. Hurink, J.L., Paulus, J.J.: Online Scheduling of Parallel Jobs on Two Machines is 2-Competitive. Operations Research Letters (to appear) doi:10.1016/j.orl.2007.06.001
5. Johannes, B.: Scheduling parallel jobs to minimize the makespan. Journal of Scheduling 9, 433–452 (2006)
6. Pruhs, K., Sgall, J., Torng, E.: Online Scheduling. In: Leung, J.Y-T. (ed.) Handbook of Scheduling: Algorithms, Models, and Performance Analysis, ch. 15, pp. 15–41. CRC Press, Boca Raton (2004)
7. Ye, D., Zhang, G.: On-line Scheduling of Parallel Jobs. In: Kralovic, R., Sýkora, O. (eds.) SIROCCO 2004. LNCS, vol. 3104, pp. 279–290. Springer, Heidelberg (2004)
8. Ye, D., Zhang, G.: On-line Scheduling of Parallel Jobs in a List. Journal of Scheduling (to appear) doi:10.1007/s10951-007-0032-x