Available online at www.sciencedirect.com
SCIENCE DIRECT®

ELSEVIER

Computers & Operations Research 32 (2005) 3265–3271

computers & operations research

www.elsevier.com/locate/cor

A quadratic time algorithm to maximize the number of just-in-time jobs on identical parallel machines

Ondřej Čepek[a,1,*], Shao Chin Sung[b]

[a]Department of Theoretical Informatics and Mathematical Logic, Charles University, Malostranské nám. 25, 1118 00 Praha 1, Czech Republic
[b]School of Information Science, Japan Advanced Institute of Science and Technology 1-1 Asahidai, Tatsunokuchi, Ishikawa 923-1292, Japan

Abstract

In this paper, we study a scheduling problem on identical parallel machines, in which a processing time and a due date are given for each job, and the objective is to maximize the number of just-in-time jobs. A job is called just-in-time if it is completed exactly on its due date. We discuss the known results, show that a recently published greedy algorithm for this problem is incorrect, and present a new quadratic time algorithm which solves the problem.

Keywords: Scheduling; Identical parallel machines; Just-in-time jobs

1. Introduction

We are concerned with a task of scheduling jobs nonpreemptively on identical parallel machines. The problem studied here can be described as follows: there is a set of n jobs, each job is assigned a processing time and a due date, and the objective is to maximize the number of jobs completed exactly on their due dates.

The single machine case is equivalent to a classical scheduling problem, sometimes called activity selection problem. In that setting, each job (activity) is usually specified by a time interval in which it must be processed, and the objective is to select as many jobs as possible so that their intervals do not

* Corresponding author. Tel.: +420-2-2191-4246; fax: +420-2-2191-4323.
E-mail address: cepek@ksi.ms.mff.cuni.cz (O. Čepek)
[1] Teaching at the Institute of Finance and Administration - VŠFS.

0305-0548/$ - see front matter
doi:10.1016/j.cor.2004.05.011

overlap. For this problem a polynomial time algorithm is long known. It appears, e.g. in the classical monograph [1, Chapter 17, "Greedy Algorithms", p. 330] as a "textbook example" of a greedy algorithm. A simple modification of this algorithm was published by Lann and Mosheiov [2], where the "just-in-time scheduling" terminology was already used. The algorithm in [1] builds the schedule from the beginning forward, while the algorithm in [2] builds the schedule from the end backward; otherwise both algorithms are identical.

Furthermore, paper [2] considered a single machine case in which for each due date an interval (time window) rather than a single number is specified, and a job is considered just-in-time if its completion falls within the given interval. A polynomial time algorithm maximizing the number of just-in-time jobs was presented there for this generalized case. This algorithm was shown to be wrong by Hiraishi et al. [3], where it was also proved that this generalized problem is NP-hard.

However, the main contribution of [3] lies in the results for the multi-machine case (identical parallel machines) with ordinary due dates (not the interval type). It was shown there, that the problem is solvable in polynomial time even if a nonnegative weight is assigned to each job, a nonnegative set-up time is assigned to each (ordered) pair of jobs, and the objective is to maximize the weighted number of just-in-time jobs. The algorithm proposed in [3] uses a min-cost-flow algorithm as its subroutine. Unfortunately, the authors specify no details regarding the complexity of the proposed algorithm, and suggest no specific min-cost-flow algorithm to use, except of saying that it can be found in the network flows monograph [4]. Although some details are not completely clear to us, it seems that even if the best min-cost-flow algorithm from [4] is used, it will cause the algorithm from [3] to have a relatively high $\Omega(n^4 \log n)$ running time, where n is the number of jobs.

In [5] Sung and Vlach show that when the number of machines is fixed, the weighted problem considered in [3] is solvable in polynomial time (exponential in the number of machines) no matter whether the parallel machines are identical, uniform, or unrelated. However, when the number of machines is a part of the input, the unrelated parallel machines case of the problem becomes strongly NP-hard.

The simplest multi-machine case, i.e. identical machines, unit weights, and no set-up times, was studied by Lann and Mosheiov [6], where a simple greedy $O(n \log n)$ algorithm was presented, greatly improving the complexity of the more general algorithm from [3]. Unfortunately, this algorithm is incorrect, as is shown in the appendix.

In this paper we present a different polynomial time algorithm for this problem, whose running time is $O(n \log n + nm)$, where m is the number of machines and n is (as before) the number of jobs. Since the problem becomes trivial for $m \geqslant n$ (each job can be scheduled just-in-time on one of the machines), we may assume $m < n$. This gives an upper bound $O(n^2)$ for the running time, which is still much better than $\Omega(n^4 \log n)$ running time of the more general algorithm from [3].

2. Maximizing the number of just-in-time jobs

The problem in which the objective is to maximize the number of just-in-time jobs can be formulated as follows: There are m parallel identical machines $M_1, M_2, \ldots, M_m$ and n jobs $J_1, J_2, \ldots, J_n$ with integer processing times $p_1, p_2, \ldots, p_n$ and due dates $d_1, d_2, \ldots, d_n$. The problem is to find a feasible nonpreemptive schedule in which the number of just-in-time jobs is maximized, where a job is called *just-in-time* if it is completed exactly on its due date. The feasibility of schedules and the objective function can be described formally as follows.