Available online at www.sciencedirect.com

SCIENCE DIRECT

Computers & Operations Research 32 (2005) 599–611

computers & operations research

www.elsevier.com/locate/dsw

Total completion time minimization in a computer system with a server and two parallel processors

S. Guirchoun[a,b,*], P. Martineau[a], J.-C. Billaut[a]

[a] Laboratoire d'Informatique, Ecole Polytechnique de l'Université de Tours, 64 av J Portalis, 37200 Tours, France
[b] Accellent, 27 rue Michael Faraday, 37170 Chambray les Tours, France

Abstract

The context of the problem tackled in this paper is a computer system composed by a single server and two identical parallel machines. The processing times on the server are assumed to be unary and the objective is to minimize the total completion time. The papers dealing with scheduling problems with a server generally consider that the setup activities require simultaneously the server and the machine. In this paper, this constraint is not considered and the studied problem is a two-stage hybrid flow shop with no-wait constraint between the two stages. An algorithm that can solve optimally this problem in $O(n \log(n))$ time is proposed. Finally, it is shown that solving this problem optimally leads to an optimal solution to the problem without the no-wait constraint.

Keywords: Scheduling; Parallel machines; Single server; Hybrid flow shop

1. Introduction

Parallel processing has received a lot of attention these last years because of its efficiency, which is a crucial part in success of parallel computer systems. Hence, there is a necessity for developing efficient scheduling algorithms in computer systems. A classical configuration is build with one server of files linked with clients by a network [1]. The first step of the parallelization consists in dispatching on a designed client the code and the data of each program that can be parallelized. For each program that has to be sent to a client, the server must, successively, read the code and data on the hard disk and send them to the client by the network. In a first approximation these two steps can be modeled by one task on the server. This assumption is not excessive because in

* Corresponding author. Tel.: +33-247-361-414; fax: +33-247-361-422.
E-mail addresses: samuel.guirchoun@etu.univ-tours.fr (S. Guirchoun), pmartineau@univ-tours.fr (P. Martineau), billaut@univ-tours.fr (J.-C. Billaut).

doi:10.1016/j.cor.2003.08.007

such a context, the designer tries to balance the processing times on different clients to obtain the best performance of the parallelized execution. Then programs and data have often similar sizes and as a consequence, reading and sending them takes similar times. The goal to achieve can be the minimization of the maximum completion time if the user is waiting for fast result. But if the goal is to use the network with the most efficiency, the objective that has to be studied is the minimization of the mean completion time or the sum of completion times.

We consider a deterministic scheduling environment with m identical parallel machines $M_1, M_2, \ldots, M_m$ and n jobs to schedule. Each job must be processed without preemption on a machine to determine. Before its processing, a job has to be loaded on a machine. This loading activity or setup activity, is performed by another special machine, called a server. After a setup, the server is available again to perform another loading activity, that is to say a single server can handle only one job at a time and can be considered like a single machine. The loading of a job must be immediately followed by its processing. In this environment, we consider that setup times require a unit time for any job. Moreover, we assume that the transfer times between the server and the machines are non-significant. The aim of this paper is to find a feasible schedule in an environment of $m = 2$ machines, and with minimum total completion time, which corresponds to the minimization of the work in process in the network.

Many results of the last few years are issued from parallel machines scheduling problems with server. Koulamas [2] proposes a beam search heuristic algorithm for a static environment with two parallel processors and a single server where the aim is to find a feasible schedule which minimizes the machine idle time resulting from the unavailability of the server. Kravchenko and Werner [3], Hall et al. [4] and Brucker et al. [5] present a lot of complexity results for these problems. Glass et al. [6] consider related models with parallel machines for which jobs are dedicated and provide algorithmic, complexity and heuristic analysis results. Kravchenko and Werner [7] propose a heuristic to minimize the sum of the completion times in the case of unit setup times and arbitrary processing times. However, the problems tackled in these papers implicitly consider production environment where the server can be a human operator, a robot or an automated guided vehicle. Hence, during the loading operation, the performing machine cannot process another job. So, the loading activity is usually considered like a multiprocessor task, that requires simultaneously the server and the machine to be performed.

In a computer system, the server that send data to machines is called a network server. During the loading activity, it is not necessary for the performing machine to be available: it can process another job. Indeed, machines have a communication coprocessor which allows them to receive server information at any time. So, in a computer system, the loading activity can be considered like a job that requires only the server to be performed.

The computer system that we consider can be seen as a two-stage hybrid flow shop or multiprocessor flow shop, with a single machine at the first stage, which is the server and m parallel machines at the second stage, with a no-wait constraint between the two stages. These problems are known to be strongly NP-hard [8] even for their preemptive version [9]. Vignier et al. [10] and Linn and Zhang [11] propose a state-of-the-art survey on hybrid flow shop scheduling problems. However, most of the problems consider the makespan as criterion. In the case of the minimization of the sum of completion times, Pinedo [12] shows that it is possible to find an optimal solution using SPT rule, when all the operations of a job have the same processing times. Unfortunately, this result is no longer valid under the hypotheses that we consider.