ON OPTIMAL SCHEDULING PROBLEMS FOR PARALLEL MACHINES WITH A SINGLE SERVER

Chang Zheng Xie
Kyoto Institute of Technology, Kyoto 606-8585, Japan
xcz@si.dj.kit.ac.jp

Jo Kuwata, Yoshiyuki Karuno, Hiroshi Kise
Kyoto Institute of Technology, Kyoto 606-8585, Japan
{Kuwata, Karuno, Kise}@ipc.kit.ac.jp

Abstract

In this paper, we deal with parallel machine scheduling problems with a single server that are generalizations of classical parallel machine problems. Immediately before processing, each job must be loaded on a machine, and after processing, each job must be unloaded from the machine, which takes a certain load and unload time respectively. All these processes have to be done by a single server which can only handle at most one job at a time. A RCPSP solver is employed for solving the problems and computational results are proposed.

Keywords: Parallel machines, Single server, Lower bound, RCPSP solver.

1. INTRODUCTION

In recent years, a great deal of research has been focused on solving scheduling problems near to real world, and many new scheduling models have been proposed and studied. Parallel machine scheduling problems with a single server have been studied by Hall, et al., (2000), Kravchenko and Werner, (1997), and Blazewicz, et al., (1999). In their models, each job consists of two operations: the set-up operation and the process operation which are handled by a single server, $m$ parallel machines, respectively. They proved the problems is strongly NP hard.

In this paper, we study such models with an additional finishing operation in which after the process operation of each job, an unload operation is considered. Our model is more realistic. An obvious example is a so-called client-server computing system in which each client (machine in our model) reads data from the server (in the set-up operation in our model), processes the data and writes the result back to the server (in the finishing operation in our model), Blazewicz, et al., (1999) study computational complexity of this problem. Another example is a robotic (manufacturing) cell which consists of machining centers (MC) and a single robot. The robot transfers each job (one at a time) from an AS/RS (automated storage and retrieval system) to a m/c and returns it to the AS/RS after it has been processed. In this case the robot plays the role of the server; the selection and loading operations correspond to the setup operation, and the unloading and sending back operations to the finishing operation in our model. In this paper we propose to employ a RCPSP (resource constrained project scheduling problem) solver which has been developed by Nonobe and Ibaraki, (1999)*. We developed lower bounds to evaluate solutions obtained by the RCPSP solver, and executed some numerical experiments.

2. PROBLEM STATEMENT

Parallel machine scheduling problems with a single server (a robot) that are addressed here are special cases of classical parallel machine problems, and may be stated as follows.

Our system consists of a set of $n$ jobs, $N = \{j_1, j_2, \ldots, j_n\}$; a set of $m$ parallel machines, $M = \{M_1, M_2, \ldots, M_m\}$; and a single server $M_s$, where each machine can process at most one job at a time. Each job consists of three operations: "head", "body" and "tail" which are handled in this order. The operation head is implemented jointly by the robot and a m/c among the $m$ machines in parallel, taking $h_j$ time units for job $j$. The operation body is implemented by the m/c alone, taking $h_j$ time units for job $j$. The operation tail is implemented jointly by the robot and the m/c, taking $t_j$ time units for job $j$. In our model, parallel machines are of two types: set $M_i$ of identical machines that are capable of processing any job, and set $M_d$ of dedicated machines that are capable of processing only certain designated jobs. The objective of the problems is to find a feasible schedule which minimizes the makespan. Fig. 1 shows a feasible schedule of an example with 3 parallel machines and 4 jobs.

*http://www-or.amu.kyoto-u.ac.jp/~nonobe
3. RCPSP SOLEVER

Many scheduling problems can be formulated as the resource constrained project scheduling problems (RCPSP) (e.g., see Brucker, et al., 1999). Nonobe and Ibaraki (2000) have developed a general purpose RCPSP algorithm that is a tabu search based heuristic, and demonstrated its superiority for various complicated scheduling problems including flow shop and job shop scheduling problems.

Our solution method is mainly based on this algorithm with some modifications.

4. LOWER BOUNDS

For the purpose of evaluating solutions obtained by the RCPSP solver we propose the following lower bounds.

4.1 Server Based Lower Bound

$$\sum_{j=1}^{n} (h_j + t_j) \leq \pi^*$$ (1)

where $\pi^*$ denotes the makespan of the optimal schedule.

4.2 Parallel-machine Based Lower Bound

$$\sum_{j=1}^{n} (h_j + b_j + t_j) x(j,M_d) \leq \pi^*$$ (2)

where

$$x(j,M_d) = \begin{cases} 1 & \text{job } j \text{ processed on a dedicated machine } M_d \\ 0 & \text{otherwise} \end{cases}$$

and $m_i$ denotes the number of identical machines.

4.3 Server and Parallel-machine Based Lower Bound

Let $h_i'(i = 1, 2, \ldots, m - 1)$ and $t_i'(i = 1, 2, \ldots, m - 1)$ be the $(m - 1)$ shortest head times and the $(m - 1)$ shortest tail times, respectively. Let

$$I_i = h_1' + h_2' + \cdots + h_{i-1}'' + I_{m-2} + h_{m-1}'''$$ (Fig. 1)

and

$$H_{\min} = \sum_{i=1}^{m-1} I_i, \quad T_{\min} = \sum_{i=1}^{m-1} t_i'$$

then

$$H_{\min} + \sum_{j=1}^{n} (h_j + b_j + t_j) \left( \frac{1}{m} \right) + \frac{T_{\min}}{2} \leq \pi^*$$ (3)

Fig. 1 A schedule for three machines with a single server processing four jobs

Fig. 2 Variation of the optimal objective value with mean processing time $b$ for the case of $m = 5, n = 30, \bar{h} + \bar{b} + \bar{t} = 100$

5. EXPERIMENTAL RESULT

The RCPSP solver by Nonobe and Ibaraki is employed to examine the problem. Fig. 2 show the experimental result with 5 identical machines (no dedicated ones) and 30 jobs of the variation of the optimal objective value by changing the mean body processing time from 10 to 90. The mean value of the sum of $h_j, b_j$ and $t_j$ is set as 100. As shown in Fig. 2, the objective function is decreased as the “body” times increases (and hence as the sum of “head” and “tail” times decrease). This means that if the load of the server is heavy, the makespan can be hardly improved. The results are compared with the lower bounds we have proposed. The differences between these two is at most 10%, thus the RCPSP solver and the lower bound proposed are effective for the problem with identical parallel machines.

References


