

Complexity Results for Parallel Machine Problems with a Single Server

Peter Brucker²

Universität Osnabrück, Fachbereich Mathematik/Informatik, 49069 Osnabrück, Germany
peter@mathematik.uni-osnabrueck.de

Clarisse Dhaenens-Flipo³

Leibniz - IMAG, 46 Avenue Felix Viallet, 38031 Grenoble Cedex, France
Clarisse.Flipo@imag.fr

Sigrid Knust²

Universität Osnabrück, Fachbereich Mathematik/Informatik, 49069 Osnabrück, Germany
sigrid@mathematik.uni-osnabrueck.de

Svetlana A. Kravchenko^{1,2,3}

Institute of Engineering Cybernetics, Surganov Str. 6, 220012 Minsk, Belarus
kravch@newman.bas-net.by

Frank Werner

Otto-von-Guericke-Universität, Fakultät für Mathematik, PSF 4120, 39016 Magdeburg, Germany
frank.werner@mathematik.uni-magdeburg.de

Abstract

We continue studying questions concerning algorithmic and computational complexity issues for parallel machine scheduling with a single server begun in [1]. The only difference between considered problems and classical parallel machine problems is that each job before processing needs to be loaded on a machine by the server. As a result the performing of any job consists of two parts, namely, the loading part and the processing part. The loading of each job j has to be performed by the server and by some machine i for s_j time units. The processing of job j is performed by the machine i for p_j time units. Each machine as well as the server can deal with only one job at a time, and each job can deal simultaneously with one machine only.

1 Introduction

The problem in question can be described as follows. There are m identical parallel machines $1, \dots, m$ which must process n jobs $1, \dots, n$. Each job j has a known integer processing time p_j and it must be loaded on a machine before its processing. The

¹Supported by Belarussian Fundamental Research Found, Project $\Phi 97 - 147$

²Supported by the Deutsche Forschungsgemeinschaft, Project 'Komplexe Maschinen-Scheduling-probleme'

³Supported by the International Association for the Promotion of Cooperation with Scientists from the Independent States of the Former Soviet Union, Project INTAS-96 - 820

loading takes s_j units of time. This loading, which is called a setup, is performed by a server. There is only one server serving all machines. Each machine as well as the server can deal with only one job at a time and each job can be processed by at most one machine at a time. We denote this problem by $P, S1 \mid \beta \mid \gamma$. In the paper we prove unary NP-hardness of problems $P2, S1 \mid p_j = 1, r_j \mid L_{max}$, $P, S1 \mid s_j = 1 \mid \sum C_j$, and $P2, S1 \mid p_j = 1, r_j \mid \sum C_j$, binary NP-hardness of problems $P2, S1 \mid p_j = p \mid C_{max}$, and $P2, S1 \mid p_j = p \mid \sum C_j$, and develop polynomial algorithms for problems $P, S1 \mid p_j = 1, r_j, s_j = 1 \mid \sum w_j T_j$, $P, S1 \mid p_j = 1, r_j, s_j = s \mid \sum w_j C_j$, $P, S1 \mid p_j = 1, r_j, s_j = s \mid \sum w_j U_j$, $P, S1 \mid p_j = 1, r_j, s_j = s \mid \sum T_j$, $P, S1 \mid p_j = p, r_j, s_j = s \mid C_{max}$, $P, S1 \mid p_j = p, r_j, s_j = s \mid \sum C_j$, $P, S1 \mid p_j = 1, r_j \mid C_{max}$, $P, S1 \mid p_j \geq m - 2, s_j = 1 \mid \sum C_j$, $P3, S1 \mid s_j = 1 \mid \sum C_j$, and $P, S1 \mid p_j = p, s_j = s \mid \sum f_j$, where f_j is a nondecreasing function. Furthermore, we present two results which seem the most interesting. The first one is a reduction of parallel machine problems to server problems and the second one is a NP-hardness proof for the problem $P, S1 \mid s_j = 1 \mid \sum C_j$.

2 Reduction from parallel machine problems

Theorem: For $\alpha \in \{m, \circ\}$, arbitrary processing times p_j , $\beta \in \{r_j, \circ\}$ and each objective function $\gamma \in \{C_{max}, L_{max}, \sum w_j C_j, \sum w_j U_j, \sum w_j T_j\}$ problem $P\alpha \mid \beta \mid \gamma$ reduces polynomially to $P\alpha, S1 \mid \beta, s_j = 1 \mid \gamma$.

Sketch of the proof: Given an instance I of problem $Pm \mid \beta \mid \gamma$, we construct an instance I' of $Pm, S1 \mid \beta, s_j = 1 \mid \gamma$ by scaling the data with a large, but polynomially bounded number a and introducing unit setup times for all jobs. A feasible schedule S for I can be transformed into a feasible schedule S' of I' by stretching it with the factor a , considering the unit setup times before processing and moving some jobs to the right if their setups overlap with others. Conversely, a feasible schedule S for I can be constructed from a feasible schedule S' for I' by applying the inverse transformation. The idea of the scaling is that the setup and idle times introduced by the server do not change the objective value too much. For the objective values of S and S' it can be shown that $\gamma(S) \leq y$ if and only if $\gamma(S') = y'$ for an appropriate value y' depending on y and γ . ■

3 NP-hardness proof

Theorem: Problem $P, S1 \mid s_j = 1 \mid \sum C_j$ is unary NP-hard.

Sketch of the proof: We prove NP-hardness by a reduction from the known unary NP-complete 3-partition problem. An instance of the 3-partition problem consists of $3k + 2$ natural numbers k , H , and a_1, \dots, a_{3k} with $H/4 < a_i < H/2$ for $1 \leq i \leq 3k$ and $\sum_{i=1}^{3k} a_i = kH$. Does there exist a partition of the set $\{1, \dots, 3k\}$ into k sets G_1, \dots, G_k of triples such that $\sum_{i \in G_j} a_i = H$ for $1 \leq j \leq k$?

Given an instance of 3-partition, create an instance of the scheduling problem with $k + 1$ machines and $n := k^2H + k + 1$ jobs. For convenience we divide all jobs into three types: There are $3k$ jobs of the first type with $s_j = 1$, $p_j = k^2a_j$, where $j = 1, \dots, 3k$. There are $k^2H - 3k$ jobs of the second type with $s_j = 1$, $p_j = 0$, where $j = 3k + 1, \dots, k^2H$. There are $k + 1$ jobs of the third type with $s_j = 1$, $p_j = k^2H^2$, where $j = k^2H + 1, \dots, k^2H + k + 1$.

The decision problem is: Does there exist a feasible schedule with $\sum C_j \leq y$, where $y = 0.5(k^2H + k + 1)(k^2H + k) + k^3H^2 + k^3H + k^2H^2 + k^2H + k + 1$?

Note that for any single server problem $\sum_j C_j = \sum_j t_j + \sum_j s_j + \sum_j p_j$ holds, where t_j is the starting time of job j . Since $\sum_j s_j = k^2H + k + 1$ and $\sum_j p_j = k^3H^2 + k^3H + k^2H^2$, for the scheduling instance we have

$$\sum_{j=1}^n C_j = \sum_{j=1}^n t_j + k^2H + k + 1 + k^3H^2 + k^3H + k^2H^2,$$

and therefore a lower bound for the value $\sum_j C_j$ is determined by $\sum_j t_j = 0 + 1 + \dots + [k^2H + k]$, i.e. we always have

$$\begin{aligned} \sum_{j=1}^n C_j &\geq 0 + 1 + \dots + [k^2H + k] + k^3H^2 + k^3H + k^2H^2 + k^2H + k + 1 \\ &= 0.5(k^2H + k + 1)(k^2H + k) + k^3H^2 + k^3H + k^2H^2 + k^2H + k + 1 \\ &= y, \end{aligned}$$

which is achievable if and only if the server works without idle times. ■

References

- [1] N. Hall, C. Potts, C. Srisukandarajah (1996) Parallel machine scheduling with a common server, The Fifth International Workshop on Project Management and Scheduling (PMS'96), Abstracts, 102-106.