Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 185 (2008) 1308–1318
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Feasible job insertions in the multi-processor-task job shop

Heinz Gröflin *, Andreas Klinkert, Nguyen Pham Dinh

Department of Informatics, University of Fribourg, Bd de Pérolles 90, CH-1700 Fribourg, Switzerland

Received 27 August 2004; accepted 31 October 2005
Available online 13 November 2006

Abstract

The Multi-Processor-Task Job Shop is an extension of the Job Shop problem where an operation of a job requires a set of machines instead of a single machine. Job insertion is the following: given a feasible schedule of $n-1$ jobs, find a feasible insertion of job n into the schedule such that makespan is minimized. The problem is known to be NP-hard already for the Job Shop case. In this note, a polyhedral description of all feasible insertions is derived, settling an open problem recently proposed by Kis and Hertz. Constrained feasible insertions, satisfying additional constraints, are introduced and a feasibility theorem is established. A lower bound on the job insertion problem is derived and computed by repeatedly invoking the feasibility theorem. Numerical results show high quality of the bounds and short computation times.

Keywords: Scheduling; Job shop; Multi-processor-task; Job insertion; Polyhedral description

1. Introduction

One of the most studied machine scheduling problems is the Job Shop scheduling problem (JS): n jobs J_j, $j=1,\dots,n$, have to be processed on a set M of machines. Job J_j consists of n_j operations O_{ij}, $i=1,\dots,n_j$, that have to be processed in sequence, O_{ij} being the ith operation of job J_j. Each operation O_{ij} is processed on a dedicated machine $m_{ij} \in M$ during a processing time $d_{ij} > 0$. All jobs are available at time 0, and at any time, a machine can process at most one operation. The problem consists in finding a schedule of the jobs that minimizes the makespan $C_{\max} = \max\{C_j : j=1,\dots,n\}$, C_j denoting the completion time of (the last operation of) job J_j.

Here we address the following generalization: for each operation O_{ij}, a *set* of machines $M_{ij} \subseteq M$ is needed for its processing. More specifically, while O_{ij} is being processed, the machines of M_{ij} are occupied simultaneously by O_{ij} during its processing time. This generalization of the Job Shop Scheduling Problem has been studied e.g. by Brucker and Krämer [1] and Brucker and Neyer [2]. The later refer to this problem as the Multi-Processor-Task Job Shop problem and we shall borrow this denomination from them and abbreviate it as MPTJS.

In this paper, we focus on the so-called Job Insertion Problem for the MPTJS. Job insertion in general consists in the following. Given a feasible schedule of $n-1$ jobs J_j, $j=1,\dots,n-1$, insert all operations of job J_n simultaneously into the

* Corresponding author.
E-mail address: heinz.groeflin@unifr.ch (H. Gröflin).

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2005.10.077

schedule. If the insertion results in a feasible schedule of the n jobs, the insertion is feasible. The Job insertion problem consists in finding a feasible insertion minimizing the resulting makespan.

A first motivation for addressing job insertion in MPTJS stems from previous research of the authors [7,3] in job shop scheduling with synchronization and blocking constraints, a type of scheduling problem which, in some sense, is more complex than the Job Shop scheduling problem JS or MPTJS. A local search approach using job insertion as a mechanism for neighbor generation was used with quite satisfying results. We refer the reader to [3] for details.

Recent research of Kis and Hertz provided further motivation for this work. In [5,6], they study job insertion for the classical Job Shop scheduling problem (JS). Although the problem is NP-hard, they are able to give a polyhedral description of the set of feasible insertions, and, based on that description, they derive strong lower bounds for the Job Insertion Problem. In their conclusion in [6], Kis and Hertz indicate as directions for future research job insertion for more general scheduling problems, (sic) "e.g. when the operations of the jobs must be processed on sets of machines instead of single machines. In this case, the characterization of the set of feasible insertions of a sequence of operations into a schedule is a major open problem". The present work proposes an answer to this problem, as well as an alternative characterization for the Job Shop case. It also provides an efficient lower bound computation for the minimum makespan job insertion problem in the MPTJS, the first bound for this problem to our knowledge.

The plan of the paper is as follows. In the next section, we first formulate the Multi-Processor-Task Job Shop problem (MPTJS) as a disjunctive program and a selection problem in a disjunctive graph, in order to then state the Job Insertion problem for the MPTJS, the main object of this paper. In Section 3, we derive a polyhedral description of the solution space of the Job Insertion problem, characterizing the family of all feasible insertions as being isomorphic to the family of all stable sets of maximum cardinality of a certain bipartite graph. Section 4 is devoted to feasible insertions satisfying some additional constraints, including preselections. After a straightforward polyhedral characterization of such constrained feasible insertions, a necessary and sufficient condition for the existence of a constrained insertion is derived, which can be tested very efficiently. Also, a decomposition result for the family of all feasible constrained insertions is established. In Section 5, a lower bound for the minimum makespan of the Job Insertion problem is derived, which is computed via bisection search and repeated application of the efficient feasibility test established in the preceding section. Extensive numerical results demonstrate the high quality of the lower bound obtained in short computation time.

2. Problem formulations

The well-known disjunctive programming formulation and disjunctive graph representation of the classical job shop problem JS is easily extended to the MPTJS.

We shall use the following definitions and notations. Graphs are directed graphs if not specified otherwise, and for any arc $e = (i,j)$ of a graph $G = (V, E)$, $t(e) = i$ is the tail and $h(e) = j$ the head of e, and for any node set N, $\delta^{-}(N) = \{e \in E: t(e) \notin N, h(e) \in N\}$ and $\delta^{+}(N) = \{e \in E: t(e) \in N, h(e) \notin N\}$.

$\mathscr{J}$ is the set of all jobs and I the set of all operations. A job $J \in \mathscr{J}$ is identified with its set of operations, i.e. $J \subseteq I$ and $\mathscr{J} \subseteq 2^{I}$ is a partition of I. For ease of notation, we denote the operations of a job J by indexing J with a subscript: the sequence of operations of job J is denoted $J_1, J_2, \ldots, J_{|J|}$. Operations i and j are *consecutive operations of* J if $i = J_r, j = J_{r+1}$ for some r, $1 \leqslant r < |J|$ and the set of these $|J| - 1$ ordered pairs (i,j) is denoted by A_J. Furthermore, let M be the set of machines, and for any operation $i \in I$, let M_i be the set of machines needed by i, and $d_i > 0$ be the processing time.

Finally, let σ and τ be dummy start and finish operations of duration zero, having to be performed before, respectively after all operations. If x_i denotes the starting time of operation $i \in I \cup \{\sigma, \tau\}$, the disjunctive programming formulation of the MPTJS is

$$\text{minimize} \quad x_\tau \tag{1}$$

$$\text{subject to}: x_j - x_i \geqslant d_i \vee x_i - x_j \geqslant d_j \quad \text{for all } \{i, j\} \in B, \tag{2}$$

$$x_j - x_i \geqslant d_i \quad \text{for all } (i,j) \in A_J,\ J \in \mathscr{J}, \tag{3}$$

$$x_i - x_\sigma \geqslant 0, \quad x_\tau - x_i \geqslant d_i \quad \text{for all } i \in I, \tag{4}$$

$$x_i, x_\sigma, x_\tau \geqslant 0 \quad \text{for all } i \in I. \tag{5}$$