Computers & Operations Research 39 (2012) 736–745

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Two-stage hybrid flow shop with precedence constraints and parallel machines at second stage

Sergiu Carpov [a,b,*], Jacques Carlier [b], Dritan Nace [b], Renaud Sirdey [a]

[a] CEA, LIST, Embedded Real Time Systems Laboratory, Point Courrier 94, 91191 Gif-sur-Yvette Cedex, France
[b] UMR CNRS 6599 Heudiasyc, Université de Technologie de Compiègne, Centre de recherches de Royallieu, BP 20529, 60205 Compiègne Cedex, France

ARTICLE INFO

Available online 2 June 2011

Keywords:
Hybrid flow shop
Precedence relations
Randomized list scheduling
Multiprocessor scheduling

ABSTRACT

This study deals with the two-stage hybrid flow shop (HFS) problem with precedence constraints. Two versions are examined, the classical HFS where idle time between the operations of the same job is allowed and the no-wait HFS where idle time is not permitted. For solving these problems an adaptive randomized list scheduling heuristic is proposed. Two global bounds are also introduced so as to conservatively estimate the distance to optimality of the proposed heuristic. The evaluation is done on a set of randomly generated instances. The heuristic solutions for the classical HFS in average are *provably* situated below 2% from the optimal ones, and on the other hand, in the case of the no-wait HFS the average deviation is below 5%.

1. Introduction

This work considers the hybrid flow shop problem under precedence constraints. More precisely the two-stage hybrid flow shop $HF(1,P_m)$ with precedence constraints at the second stage is studied, by abuse of notation we denote it HFS in what follows. Assume a set of n jobs has to be processed in two stages. There is only one machine for the first stage and m identical parallel machines for the second stage. Each job $i \in \{1,\ldots,n\}$ consists of two operations: the first operation of duration $a_i > 0$ is executed at the first stage, and afterwards the second operation of duration $b_i > 0$ is executed at the second stage. No preemption is allowed in operation execution. The precedence constraints of the operations at the second stage are given by a directed acyclic graph $G=(V,E)$, where V represents the set of jobs and E gives the dependence relations between those jobs. There are no precedence constraints between the operations at the first stage.

The objective is to minimize the *maximum completion time* or *makespan*. Two different cases of HFS can be distinguished: the *no-wait* HFS when once a job has started it is executed on all the stages without being interrupted (the end time of the first stage operation coincides with the start time of the second stage operation) and the *classical* HFS when no such constraint is imposed. In the $\alpha|\beta|\gamma$ notation the flow shop problems we examine are $HF(1,P_m)|G_1=\emptyset, G_2=G|C_{max}$ and $HF(1,P_m)|G_1=\emptyset, G_2=G, \text{no}-\text{wait}|C_{max}$.

Despite that no precedence relations are defined for the first stage operations, the second stage constraints can be extended over the first stage because they are dominating the order in which the first stage operations are executed. This fact is obvious in the case of no-wait HFS. On the other hand it can be easily shown that for any given solution in a classical HFS, rescheduling the first stage operations following the same second stage schedule does not change the solution value. Hence, in what follows we consider that if a second stage operation must be executed after another second stage operation then the corresponding first stage operations must follow the same order.

A practical application of the HFS problem arises in modeling the execution of an algorithm on a parallel computer. Each algorithm task can be viewed as two consecutive operations, the first one is the loading of the data used by the task from the external memory and the second one is the task execution itself. Usually in a parallel computer the memory accesses are done sequentially, so only one data loading can be done at a time, whereas the execution of the tasks can be done concurrently on the available processors. Hence the data loading corresponds to the first stage operation in the HFS problem, and the task execution corresponds to the second stage operation. Second stage precedence relations between the operations are equivalent to the partial order of algorithm tasks and reflect the internal data dependencies (amongst other dependencies). In order to limit the data buffering, the execution of a task has to start when its data loading is finished, this corresponds to the no-wait case of the

* Corresponding author at: CEA, LIST, Embedded Real Time Systems Laboratory, Point Courrier 94, 91191 Gif-sur-Yvette Cedex, France. Tel.: +33 1 69 08 60 48.
E-mail addresses: sergiu.carpov@cea.fr (S. Carpov), carlier@hds.utc.fr (J. Carlier), dnace@hds.utc.fr (D. Nace), renaud.sirdey@cea.fr (R. Sirdey).

0305-0548/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2011.05.020

HFS, whereas the classical HFS corresponds to the case when no space limit is imposed on the data buffering.

The paper is organized as follows: after a brief description of related works in Section 2, two global lower bounds are introduced in Section 3. Section 4 presents a list scheduling heuristic, and, in Section 5 we describe a randomized version of this algorithm. In Section 6 the lower bounds and heuristics performances are compared using randomly generated instances and Section 7 concludes the paper.

2. Related works

The literature on the hybrid flow shop problem under precedence constraints is quite scarce, even though a lot of work exist on the hybrid flow shop and on the flow shop with precedence relations. For a review of the plentiful work on the hybrid flow shop problem we refer to [1,2]. We shall note that most of the work is done for the general m-stage hybrid flow shop, nevertheless many authors tried to adapt the Johnson algorithm for the two-stage flow shop. A model close to ours, the two-stage hybrid flow shop with parallel machines at first stage only is studied in [3]. The authors determine the optimal ordering at the second stage given a scheduling of jobs on first stage and introduce some interesting lower bound concepts.

Although less represented in the literature, the flow shop problem under precedence constraints is quite well studied. In [4] the authors provide a classification of two and three machine flow shop problems under machine-dependent precedence constraints. Different models of shop scheduling problems with precedence constraints are considered in [5]. In their study the authors introduce two types of precedence constraints and provide complexity results and some polynomial time algorithms for shop scheduling models. The authors of [6] propose to reduce the job shop problem to a flow shop problem under precedence constraints, and introduce several modified flow shop heuristics for solving the flow shop problem constrained by precedence relations.

The hybrid flow shop problem under precedence constraints is studied in a few papers [7–9,1], from an applicative point of view. In the studies mentioned above some heuristics are proposed. The authors are using stage-independent precedence relations between the jobs and different optimization criteria.

3. Lower bounds

Without loss of generality we suppose, in what follows, that the digraph $G=(V,E)$ describing the precedence relations between the operations at the second stage contains one source vertex, denoted 0, and one sink vertex, denoted $*$, with zero processing times. Also we suppose that the number of jobs is greater than the number of available second stage machines, $n>m$.

3.1. Global lower bound 1

Some concepts of the following lower bound were introduced in [10] for the hybrid flow shop problem. We have adapted it in order to take advantage of the second stage precedence relations.

$$GLB1=\max(GLB1^1,GLB1^2)$$

In the first part $GLB1^1$ of the bound we take into account that there is inevitably an idle time at the second stage machines during the execution of the first $m+1$ jobs. During this idle time the first stage operations of the respective jobs are executed (see Fig. 1 for an illustration).

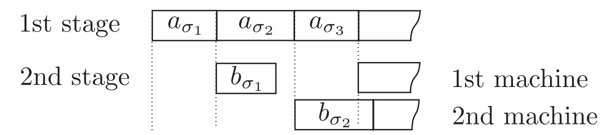

Fig. 1. Second stage idle time needed to execute first stage operations (in this example the total idle time equals to $(a_{\sigma_1}+a_{\sigma_2}+a_{\sigma_3}-b_{\sigma_1})+(a_{\sigma_1}+a_{\sigma_2})$).

Let $\sigma_1,\ldots,\sigma_{m+1}$ be the ordering of the first executed $m+1$ jobs at the first stage, σ_i represents the job in position i. For any precedence constraint between two jobs i,j, thus any edge $(i,j)\in E$ of graph G, if both jobs i,j belong to the ordering then relation $\sigma_i^{-1}<\sigma_j^{-1}$ must be satisfied (σ_i^{-1} is the position of job i). The precedence relations can be rephrased as: operation σ_1 has to be a successor of the source node 0 such that σ_1 has only one predecessor (which is the source node itself), operation σ_k must satisfy $\text{pred}(\sigma_k)\subseteq\{0,\sigma_1,\ldots,\sigma_{k-1}\}$, and so on. Here $\text{succ}(i_1,\ldots,i_k)$, $\text{pred}(i_1,\ldots,i_k)$, represents the union of successors, respectively, predecessors, of vertices $i_1,\ldots,i_k$ in the graph G.

The idle time at the second stage machine where job σ_k is executed is at least $\sum_{i=1}^{k}a_{\sigma_i}+\max(\sum_{i=k+1}^{m+1}a_{\sigma_i}-b_{\sigma_k},0)$. For the ordering $\sigma_1,\ldots,\sigma_{m+1}$ the total second stage idle time is

$$Z_1=\sum_{k=1}^{m}\left(\sum_{i=1}^{k}a_{\sigma_i}+\max\left(\sum_{i=k+1}^{m+1}a_{\sigma_i}-b_{\sigma_k},0\right)\right)$$

The sum between the minimum possible idle time Z_1 and the total amount of the second stage jobs duration divided by the number of available second stage machines gives a lower bound on the execution time. As all processing times are integers the lower bound should have also an integer value, a ceiling operator $\lceil\,\rceil$ is used for this purpose:

$$GLB1^1=\left\lceil\frac{1}{m}\left(Z_1+\sum_{i=1}^{n}b_i\right)\right\rceil$$

In order to find the sequence $\sigma_1,\ldots,\sigma_{m+1}$ which satisfies the precedence constraints and minimizes Z_1, the following combinatorial problem must be solved:

$$Z_1=\text{Minimize}\quad\sum_{k=1}^{m}\left(\sum_{i=1}^{k}a_{\sigma_i}+\max\left(\sum_{i=k+1}^{m+1}a_{\sigma_i}-b_{\sigma_k},0\right)\right)$$
$$\text{s.t.}\quad \text{pred}(\sigma_k)\subseteq\{0,\sigma_1,\ldots,\sigma_{k-1}\}$$

The following relaxation makes this problem solvable in polynomial time (here relation $\text{anc}(l)$ gives the ancestor vertices of vertex l):

$$Z_1'=\underset{\sigma_k^1}{\text{Minimize}}\sum_{k=1}^{m}a_{\sigma_k^1}(m-k+1)$$
$$+\underset{\sigma_k^2}{\text{Minimize}}\sum_{k=1}^{m}\max\left(\sum_{i=k+1}^{m+1}a_{\sigma_k^1}-b_{\sigma_k^2},0\right)$$
$$\text{s.t. } |\text{anc}(\sigma_k^l)|\le k,\quad l=1,2$$

The relaxation consists in minimizing the two parts of the objective function separately. First, an ordering σ^1 that minimizes the left hand side of Z_1' and afterwards a new ordering σ^2 which minimizes the right hand side of objective, should be found. The solution of the relaxed problem can be used for lower bound calculation in place of the initial problem solution because $Z_1'\le Z_1$. Algorithm 1 finds the solution Z_1' of the relaxed problem. We shall note that in our experiments, we have obtained a deviation between the optimal global lower bound (calculated using Z_1) and the relaxed version (calculated using Z_1') less than 0.2%. This fact indicates that there is no much benefit from using