Computing 45, 369–375 (1990)

Computing

Job-Shop Scheduling with Multi-Purpose Machines

P. Brucker and **R. Schlie**, Osnabrück

Received December 13, 1989; revised July 25, 1990

Abstract — Zusammenfassung

Job-Shop Scheduling with Multi-Purpose Machines. Consider the following generalization of the classical job-shop scheduling problem in which a set of machines is associated with each operation of a job. The operation can be processed on any of the machines in this set. For each assignment μ of operations to machines let $P(\mu)$ be the corresponding job-shop problem and $f(\mu)$ be the minimum makespan of $P(\mu)$. How to find an assignment which minimizes $f(\mu)$? For problems with two jobs a polynomial algorithm is derived.

AMS Subject Classification: 90 B 35

Key words: Job-Shop scheduling, flexible manufacturing, shortest path

Scheduling-Probleme in Job-Shops mit Mehrzweckmaschinen. Folgende Verallgemeinerung des klassischen Job-Shop Scheduling Problems wird untersucht. Jeder Operation eines Jobs sei eine Menge von Maschinen zugeordnet. Wählt man für jede Operation genau eine Maschine aus dieser Menge aus, so erhält man ein klassisches Job-Shop Problem, dessen minimale Gesamtbearbeitungszeit $f(\mu)$ von dieser Zuordnung μ abhängt. Gesucht ist eine Zuordnung μ, die $f(\mu)$ minimiert. Für zwei Jobs wird ein polynomialer Algorithmus entwickelt, der dieses Problem löst.

1. Introduction

Consider the following generalized job-shop scheduling problem. We have r jobs $J_1, \ldots, J_r$ and m different machines $M_1, \ldots, M_m$. Each job J_i consists of a number of n_i operations $O_{i1}, \ldots, O_{in_i}$ which have to be processed in this order. Associated with each operation O_{ij} there is a set $\mathcal{M}_{ij} \subseteq \{M_1, \ldots, M_m\}$ of machines and a processing time p_{ij}. O_{ij} has to be processed on exactly one machine in $\mathcal{M}_{ij}$ without preemption. No machine can process more than one job at the same time. An **assignment** μ associates with each operation O_{ij} an unique machine $\mu(O_{ij}) \in \mathcal{M}_{ij}$. A **$\mu$-schedule** is defined by the finish times C_{ij} of all operations O_{ij}. Such a schedule is **feasible** if

(i) $C_{iv} \leq C_{i,v+1} - p_{i,v+1}$ for $i = 1, \ldots, r$; $v = 1, \ldots, n_i - 1$
(ii) $\mu(O_{ij}) \neq \mu(O_{kl})$ for all $O_{ij} \neq O_{kl}$ with

$$[C_{ij} - p_{ij}, C_{ij}) \cap [C_{kl} - p_{kl}, C_{kl}) \neq \varnothing$$

(i.e. if the processing intervals of different operations overlap they must be processed on different machines).

A μ-schedule $C = (C_{ij})$ is **optimal** if it minimizes the **makespan**

$$f_\mu(C) = \max_{i=1}^{r} C_{in_i}.$$

Let $f(\mu)$ be the minimal makespan. We are interested in finding an assignment which minimizes $f(\mu)$. A problem of this type is called **job-shop scheduling problem with multi-purpose machines** (MPM-job-shop problem). This type of problem arises in connection with flexible manufacturing systems where the machines are equipped with different tools.

Notice that the MPM-job-shop is a generalization of the classical job-shop in which we have only one-element sets $\mathcal{M}_{ij}$.

In this paper we will present a polynomial time algorithm for the MPM-job-shop problem with two jobs. This algorithm is based on a graphical method for solving the job-shop problem with two jobs. A description of this graphical method can be already found in Akers [1]. It was improved and generalized by others (see Swarc [5], Hardgrave and Nemhauser [3], Brucker [2]).

In Section 2 we will review this graphical method. In Section 3 an algorithm for the MPM-job-shop problem with two jobs is developed. Some possible applications and extensions are discussed in the last section.

2. Job-Shop Scheduling Problems with Two Jobs

Assume that we have two jobs J_1 and J_2 and that an assignment μ is given. Then this problem may be formulated as a shortest path problem in the plane with rectangular objects as obstacles. The obstacles are of the form $I_{1i} \times I_{2j}$ with $\mu(O_{1i}) = \mu(O_{2j})$ where $I_{1i}(I_{2j})$ are consecutive intervals of length $p_{1i}(p_{2j})$ on the x-axis (y-axis). The paths go either diagonally or parallel to the axes and have to avoid the interior of the obstacles (see Fig. 1).

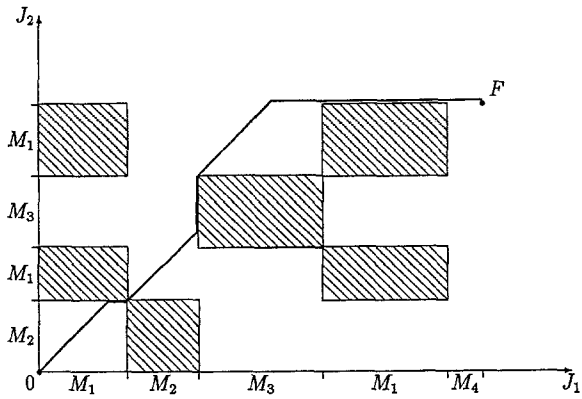

Figure 1

This shortest path problem can be reduced to the problem of finding a shortest path from an origin to some destination in a network $N(\mu) = (V(\mu), A(\mu), l(\mu))$ where

(a)

$$V(\mu) = \{O, F\} \cup \{\text{north-west corners of obstacles}\} \cup \{\text{south-east corners of obstacles}\}$$

(b) Each vertex i has at most two immediate successors which are calculated as follows. Assume R is the obstacle we hit first if we go diagonally starting in i. Then the north-west corner and south-east corner of R are immediate successors of i. If we do not hit any obstacle then F is the immediate successor of i.

(c) O is the origin and F is the destination (see Fig. 1).

(d) The length $l_{ij}(\mu) = l_{ij}$ of arc (i,j) is defined by $l_{ij} = \max\{p_{ij}^x, p_{ij}^y\}$ where p_{ij}^x and p_{ij}^y are the lengths of the two projections of the line segment connecting i and j.

In Brucker [2] it is shown, that $N(\mu)$ can be constructed in $O(n \log n)$ time where n is the number of obstacles. Furthermore the shortest path can be calculated in $O(n)$ steps. Notice that $n = O(n_1 n_2)$.

3. The MPM-Job-Shop Problem with Two Jobs

In this section we will show that also the general MPM-job-shop scheduling problem reduces to a shortest path problem in a network $N = (V, A, l)$ with a polynomial number of vertices.

3.1. The Network

Let $\mathcal{A}$ be the set of all possible assignments μ. Then we define the network $N = (V, A, l)$ by setting

$$V = \bigcup_{\mu \in \mathcal{A}} V(\mu), \quad A = \bigcup_{\mu \in \mathcal{A}} A(\mu).$$

l_{ij} is defined as before.

Theorem 1: *A shortest path from* O *to* F *in* N *corresponds with an optimal solution of the MPM-job-shop problem.*

Proof: If μ^* is an optimal assignment then an optimal solution of the problem corresponds with some path from O to F in $N(\mu^*)$. Furthermore, if we have an arbitrary path p from O to F in N, then an assignment μ can be defined such that p is a path in $N(\mu)$. To see this let (i,j) be an arc in p and assume that i is a north-west corner of an obstacle and j is a south-east corner of an obstacle with respect to some assignment μ'. This situation is shown in Fig. 2.

Then we set $\mu(O_{1v}) = \mu'(O_{1v})$ for $v = r, r+1, \ldots, k$ and $\mu(O_{2v}) = \mu'(O_{2v})$ for $v = t, t+1, \ldots, l$. All other cases are treated similarly. The assignment μ is well defined and has the desired property. □

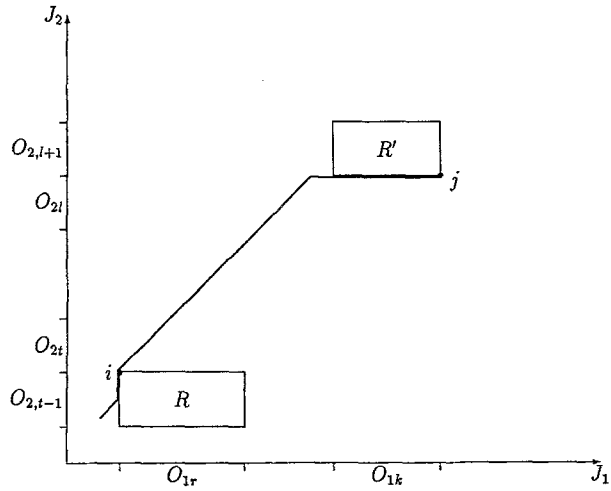

Figure 2

3.2. *Constructing the Network*

The set V consists of all north-west and south-east corners of potential obstacles. A **potential obstacle** is a rectangle $I_{1i} \times I_{2j}$ with $\mathcal{M}_{1i} \cap \mathcal{M}_{2j} \neq \emptyset$. O and F are regarded as degenerated potential obstacles where the north-west and south-east corner coincide. There are at most $n_1 \cdot n_2$ potential obstacles which can be found in time $s_1 \cdot s_2$ where

$$s_i = \sum_{v=1}^{n_i} |\mathcal{M}_{iv}| \qquad (i = 1, 2).$$

To construct A we have to find the set $S(i)$ of all immediate successors for each vertex i. Assume that i is the north-west corner of some potential obstacle R. Then we have to find the **first unavoidable obstacle R' with respect to i**, i.e. an potential obstacle R' with the property that

- there exists an assignment μ such that R' is hit first by the diagonal line l starting in i
- the length of the line segment s between i and the point in which l hits R' is as large as possible.

Let $R = I_{1r} \times I_{2,t-1}$ and $R' = I_{1k} \times I_{2,l+1}$. Then $S(i)$ consists of all north-west and south-east corners of potential obstacles $I_{1i} \times I_{2j}$ which correspond to the overlapping intervals I_{1i} and I_{2j} $(t \leq i \leq l+1; r \leq j \leq k)$ which have the additional property, that $\mathcal{M}_{1i} \cap \mathcal{M}_{2j} \neq \emptyset$ (see Fig. 3).

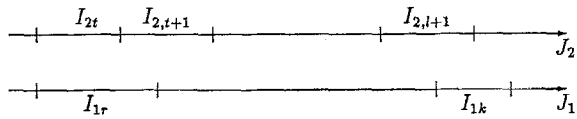

Figure 3

Thus it is easy to find all immediate successors of i if the first unavoidable obstacle is known. If i is a south-east corner of some potential obstacle R then R' and $S(i)$ are defined similarly.

Notice that each vertex i has at most $O(n)$ successors where $n = \max\{n_1, n_2\}$. Therefore the network N has at most $O(n^3)$ arcs.

3.3. *Finding the First Unavoidable Obstacle*

We now want to describe a procedure for finding the first unavoidable obstacle if we start at some vertex i. Let us assume without loss of generality that i is the north-west corner of $I_{1r} \times I_{2,t-1}$ (see Fig. 2). Then we scan the intervals I_{ij} shown in Fig. 3 in an order of nondecreasing finish times of the corresponding operations O_{ij}. During this scan the following operations are applied to the intervals I_{ij}.

- if $i = 1$ ($i = 2$) and $\mathcal{M}_{ij}$ contains only one machine M then M is deleted from all sets $\mathcal{M}_{2k}$ ($\mathcal{M}_{1k}$) with I_{2k} (I_{1k}) overlaps I_{ij}.
- if $\mathcal{M}_{ij} = \varnothing$ then I_{ij} together with the first intervall I_{lk} with $I_{ij} \cap I_{lk} \neq \varnothing$ and $|\mathcal{M}_{lk}| = 1$ defines the first unavoidable obstacle.

In Fig. 4 an example for this **forwardscan** is shown.

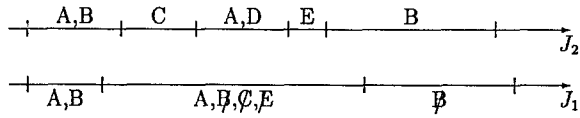

Figure 4

3.4. *Finding an Assignment*

Applying the forwardscan we get from a vertex i of an obstacle R to the next unavoidable obstacle R'. The new sets $\mathcal{M}_{ij}$ after the scan are called **modified sets**. A (partial) assignment which enables us to get from i to R' diagonally may be constructed by a **backwardscan** using the modified sets $\mathcal{M}'_{ij}$.

Starting with I_{2l} or $I_{1,k-1}$ (see Fig. 2) we scan the intervals I_{ij} of Fig. 3 in an nonincreasing order of the finish times of the corresponding operations O_{ij}. During the scan the following operations are applied to the intervals I_{ij}.

- if $i = 1$ ($i = 2$) and $\mathcal{M}'_{ij}$ contains only one machine M then M is deleted from all sets $\mathcal{M}'_{2k}$ ($\mathcal{M}'_{1k}$) with the property that I_{2k} (I_{1k}) overlaps I_{ij}. Furthermore we set $\mu(O_{ij}) := M$.
- if $\mathcal{M}'_{ij}$ contains more than one machine we arbitrarily choose one machine $M \in \mathcal{M}'_{ij}$ and set $\mu(O_{ij}) := M$.

If we apply the backwardscan to the modified sets in Fig. 4 we get the assignment shown in Fig. 5 by the encircled machines.

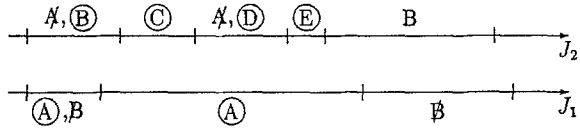

Figure 5

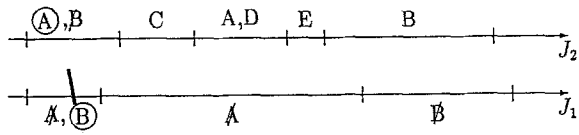

Figure 6

Notice that we may get stuck if we try to construct an assignment by a forwardscan through the modified sets. This is demonstrated in Fig. 6.

To prove that the backwardscan always works we have to show that by deletion of additional machines during the scan we never get an empty set.

Assume that during the backwardscan we reach a situation in which we have two overlapping intervals I_{1i} and I_{2j} with $\mathcal{M}'_{1i} = \mathcal{M}'_{2j} = \{M\}$. Furthermore assume that I_{1i} is scanned before I_{2j} during this scan. In this case I_{2j} was scanned before I_{1i} during the forwardscan and so M must have been deleted from the set $\mathcal{M}'_{1i}$ which is a contradiction.

3.5. The Algorithm

The algorithm can now be summarized as follows.

1. Construct the network $N = (V, A, l)$
2. Calculate a shortest path p from O to F in N.
3. Calculate an assignment corresponding with p by backwardscanning through p.

Steps 1 and 2 can be combined. Starting from O we may construct only that part of the network which is relevant. Simultanously we do the shortest path calculation. This is possible due to the fact that N is acyclic.

If we do not count the work for doing operations on the sets $\mathcal{M}_{ij}$ the complexity of the algorithm is $O(n^3_{max})$ where $n_{max} = \max\{n_1, n_2\}$.

4. Concluding Remarks

We have shown that a well known graphical method for solving the job-shop scheduling problem with two jobs generalizes to job-shops with multi-purpose machines. By reducing the MPM-job-shop problem with two jobs to a shortest path

problem we were able to find a schedule which minimizes the makespan in an efficient way. If modified slightly the method also works for arbitrary objective functions which depend monotonely on the finish times of the two jobs.

Unfortunately the methods does not lead to efficient algorithms for problems with three or more jobs. It has recently been shown by Sotskov [4] that the job-shop problem with three jobs is NP-hard. However we suspect that for a fixed number of jobs the MPM-job-shop problem is pseudopolynomially solvable.

Due to the importance of the MPM-job-shop problems for flexible manufacturing systems further research efforts should concentrate on the development of heuristics and branch and bound methods. In the latter case we may develop lower bounds by considering two-job relaxations. The classical methods of deriving lower bounds from one-machine relaxations are not applicable to the MPM-job-shop problem.

References

[1] Akers, S. B., A graphical approach to production scheduling problems. Operations Research *4*, 244–245 (1956).
[2] Brucker, P., An efficient algorithm for the job-shop problem with two jobs, Computing *40*, 353–359 (1988).
[3] Hardgrave, W. H., Nemhauser, G. L., A geometric model and a graphical algorithm for a sequencing problem. Operations Research *11*, 889–900 (1963).
[4] Sotskov, Y. N., The complexity of shop-scheduling problems with two or three jobs. European Journal of Operations Research (in press).
[5] Szwarc, W., Solution of the Akers-Friedman scheduling problem. Operations Research *8*, 782–788 (1960).

Peter Brucker
Rainer Schlie
Fachbereich Mathematik/Informatik
Universität Osnabrück
Albrechtstrasse 28
D-4500 Osnabrück
Federal Republic of Germany