European Journal of Operational Research 109 (1998) 620–631

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

Theory and Methodology

Flowshop scheduling with identical jobs and uniform parallel machines

Maged M. Dessouky [a,*], Mohamed I. Dessouky [b,1], Sushil K. Verma [a,2]

[a] *Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, CA 90089-0193, USA*
[b] *Department of Industrial Engineering, Northern Illinois University, DeKalb, IL 60115-2854, USA*

Received 1 August 1996; accepted 1 May 1997

Abstract

The single-stage scheduling problem to minimize the makespan of identical jobs on uniform parallel machines is known to be solvable in polynomial-time. We extend this work to consider multi-stage systems with flowshop configuration. We show that the 2-stage problem may also be solved in polynomial-time and for the number of stages greater than two, the problem is shown to be NP-hard. We present a branch and bound procedure which provides an optimal solution to the 3-stage problem, and a fast heuristic procedure that is shown to provide good approximate solutions on sample problems. This heuristic is a natural extension of the 2-stage polynomial-time procedure. We develop theoretical bounds showing that the maximum deviation between the solution derived by the heuristic procedure and the optimal solution is bounded by the maximum processing time of a machine at the second stage, independent of the number of jobs and the processing times at the first and third stages. We also show that the heuristic provides an approximate solution bounded by a ratio of 1.75 to the optimal solution. © 1998 Elsevier Science B.V. All rights reserved.

Keywords: Scheduling; Flowshops; Uniform parallel machines; Makespan

1. Introduction

It is common in manufacturing systems for a batch to consist of identical jobs that have similar processing requirements. We will consider the scheduling of multi-stage flowshops with identical jobs where each stage may have uniform parallel machines. The objective will be to minimize the makespan, the completion time of the last job in the batch at the last stage. This configuration was motivated by a study of a garment manufacturer. The garment in question was trousers, and the manufacturing process consisted of cutting the cloth, sewing, and attaching loops and zippers. The operations were performed by teams of seamstresses, where each team performed one operation. The operators within each stage differed in productivity, but units of trousers in the same batch were identical. This configuration is also applicable to some capital intensive industries such

* Corresponding author. Fax: 1 213 740 1120; e-mail: maged@rcf.usc.edu.
[1] Fax: 815-753-0823; e-mail: dessouky@ceet.niu.edu.
[2] Fax: 213-740-1120; e-mail: verma@rcf.usc.edu.

as semiconductor manufacturing where it is common to find newer, more modern machines running side by side with older, less efficient machines which are kept in operation because of high replacement cost. In this case, the different machines could be processing identical products. The problem is then to determine the schedule of each job on the machines to minimize the total time to complete the batch of identical jobs (makespan).

Our problem can be formally stated as follows. A set of independent jobs $J_j (j = 1, \ldots, n)$ have to be scheduled on a set of parallel machines $M_{ik} (i = 1, \ldots, m_k)$ at each operation (stage) k for $k = 1, \ldots, q$. The processing time of a job on machine $M_{ik} (i = 1, \ldots, m_k; k = 1, \ldots, q)$ is p_{ik}. Note that the processing time is only a function of the machine processing the job and we make no assumptions regarding the relationships among the processing times of machines at the same stage or across stages. We denote C_{jk} as the completion time of job J_j at stage k, $j = 1, \ldots, n$ and $k = 1, \ldots, q$. The makespan $C_{\max} = \max_{j=1,\ldots,n} C_{jq}$. We define $C^{\mathrm{o}}_{\max}$ as the optimal makespan. We use the classification scheme presented by Graham et al. (1979) to formally state the problem as $F_q(Q_1, \ldots, Q_q)|p_j = 1|C_{\max}$.

Multi-stage flowshops with identical parallel machines at each stage are often referred to in the literature as *hybrid flowshops* (Gupta, 1988). Whereas our interest is in the identical job and uniform parallel machine environment, the literature on hybrid flowshops has focused on non-identical jobs and identical parallel machines. The problem of scheduling non-identical jobs on identical parallel machines to minimize the makespan is NP-hard even for a single-stage (Garey and Johnson, 1979). For an excellent review of the single-stage problem the reader is referred to a paper by Cheng and Sin (1990). With more than a single-stage, the research has focused on developing heuristics (see e.g., Guinet and Solomon, 1996; Gupta Tunc, 1994; Lee and Vairaktarakis, 1994; Gupta, 1988).

Lawler et al. (1982) show that the 1-stage problem to minimize the makespan with identical jobs and uniform parallel machines may be solved in polynomial-time. In this paper, we show that the 2-stage problem is also polynomial. With the number of stages greater than two, the problem is shown to be NP-hard. For the 3-stage problem, an optimal solution procedure is developed based on decomposing the problem into three single-stage problems where the first and third stage problems may be solved in polynomial-time and the second stage problem, being NP-hard, is solved by a branch and bound procedure. We then theoretically and experimentally compare a fast polynomial heuristic solution procedure with the optimal solution. The heuristic is based on solving a series of single-stage problems. We show that the maximum deviation between the solution derived by the heuristic procedure and the optimal solution is bounded by the maximum processing time at the second stage, independent of the number of jobs and the processing times at the first and third stages. We also show that the heuristic provides a 1.75-approximation to the optimal solution, meaning that in the worst case, the ratio of the heuristic solution to the optimal solution is bounded by 1.75. We conclude the paper by generalizing the heuristic procedure to any arbitrary number of stages.

2. The single-stage problem

We review the main single-stage results and extend these results to introduce additional properties of the single-stage problem that will be used in solving the multiple-stage problem. Dessouky et al. (1990) proposed an $O(n \log m)$ priority queue procedure for solving the single-stage problem $Q|p_j = 1|C_{\max}$. The procedure is based on the earliest completion time (ECT) rule. Since the jobs are identical, the rule is to pick a job in sequential order and schedule it on the machine that completes it the earliest. This procedure returns the job completion times in non-decreasing order. Next, we state two key properties of this procedure, the first of which is proven by Dessouky et al. (1990) and the second follows directly.

Property 1: The ECT rule will provide a schedule in which the *minimality property* of completion times holds. This property asserts that no improvement can be obtained on the completion time of the jth job in the sequence, where $j = 1, \ldots, n$.

Property 2: The difference between the completion times of the last jobs on any two machines is bounded by the maximum processing time of any machine.

Dessouky et al. (1990) also present an $O(n \log n)$ procedure which is analogous to the ECT rule in reverse time in order to solve the case of unequal release times to minimize the makespan $(Q|r_j, p_j = 1|C_{\max})$. The procedure is based on the latest start time (LST) rule and is summarized as follows. Assume an arbitrary due date d by which all jobs must be completed. Let the latest start time of the jth job to start which allows the completion of all jobs by time d, be S_j, $j = 1, \ldots, n$. The job to machine assignments are determined by picking the jobs in sequential order and scheduling the selected job on the machine that starts processing it the latest. Note that this procedure returns the latest start times $(S_1, \ldots, S_n)$ as an ordered set. The values of r_j, $j = 1, \ldots, n$, are then matched in non-decreasing order with the latest start times. The jobs are then scheduled forward with consideration of their release times, r_j, and their machine assignments to determine their completion times, C_j, $j = 1, \ldots, n$, and $C_{\max} = \max_{j=1,\ldots,n} C_j$. We now state three key properties of this procedure, where the first two properties make use of the symmetry between the ECT and LST rules and the third follows intuitively.

Property 3: Given an arbitrary due date, d, the LST rule will provide a schedule in which the *maximality property* of latest start times holds. This property asserts that no increase can be obtained in the latest start time of the jth job in the sequence, where $j = 1, \ldots, n$.

Property 4: The difference between the latest start times of the first jobs on any two machines is bounded by the maximum processing time of any machine.

Property 5: An optimal solution to $Q|r_j, p_j = 1|C_{\max}$ exists for which all the jobs on a machine are sequenced in a non-decreasing order of their release times.

In order to state two new single-stage problems that will be used in the multiple-stage problem, we introduce the concepts of machine *availability* and *off* times. The availability time, a_i, is the earliest time machine M_i, $i = 1, \ldots, m$, can start processing any job. Conversely, the off time, f_i, is the latest time machine M_i can finish processing any job, that is, machine M_i is unavailable for processing after time f_i. In using the ECT rule to solve $Q|p_j = 1|C_{\max}$, initially $a_i = 0$, $i = 1, \ldots, m$, and they are updated every time a job is assigned to a machine. Similarly, in using the LST rule to solve $Q|r_j, p_j = 1|C_{\max}$, initially $f_i = d$, $i = 1, \ldots, m$, and they are updated every time a job is assigned to a machine.

Since, during the application of the ECT rule, a_i is the earliest completion time of the last job to finish processing on machine M_i, we extend Property 2 to Property 6 as follows.

Property 6: During any step of ECT, $|a_i - a_{i'}| \leqslant \max_{l=1,\ldots,m} p_l$ for any i and i'.

Property 7: Under ECT, the machine allocations are invariant under the transformation $a_i \leftarrow a_i + b$ for all i and for any b.

The analogous properties for LST are the following:

Property 8: During any step of LST, $|f_i - f_{i'}| \leqslant \max_{l=1,\ldots,m} p_l$ for any i and i'.

Property 9: Under LST, the machine allocations are invariant under the transformation $f_i \leftarrow f_i - b$ for all i and for any b.

We now state two new problems that will be used in solving the multiple-stage problem. Let problem $Q|a_i, p_j = 1|C_{\max}$ be the single-stage identical jobs uniform parallel machine problem to minimize the makespan where each machine is initially available for processing at time a_i, $i = 1, \ldots, m$ and let problem $Q|f_i, r_j, p_j = 1|C_{\max}$ be the case where the job release times are unequal and the machines are unavailable for processing after time f_i, $i = 1, \ldots, m$.

Lemma 1. *Application of the ECT rule to solve* $Q|a_i, p_j = 1|C_{\max}$ *will yield the minimality property for the completion times of all jobs.*

Proof. No reduction in the completion time of any job is achieved by a reassignment to another machine. □

Lemma 2. *Problem $Q|f_i, r_j, p_j = 1|C_{\max}$ is solved by the LST rule.*

Proof. Problem $Q|f_i, r_j, p_j = 1|C_{\max}$ is exactly symmetrical to $Q|a_i, e_j, p_j = 1|C_{\max}$ where the machines are initially available at times a_i and the jobs have a deadline at times e_j, by which they *must* be completed. For simplicity, we will instead prove that the ECT rule solves the problem $Q|a_i, e_j, p_j = 1|C_{\max}$. According to Lemma 1, the ECT rule provides a solution to the problem $Q|a_i, p_j = 1|C_{\max}$ which satisfies the minimality property with respect to the completion times. Therefore, problem $Q|a_i, e_j, p_j = 1|C_{\max}$ has a feasible solution if and only if the optimal completion times of problem $Q|a_i, p_j = 1|C_{\max}$ are feasible with respect to the deadlines e_j. Moreover, if problem $Q|a_i, e_j, p_j = 1|C_{\max}$ is feasible then the optimal solution to $Q|a_i, p_j = 1|C_{\max}$ is also optimal to the former problem where the optimal completion times of the latter problem are matched to the deadlines in non-decreasing order. □

3. The two-stage problem

We first state two lemmas that form the basis of the solution procedure to the 2-stage problem.

Lemma 3. *There exists an optimal solution (C^*_{jk}, $j = 1, \ldots, n$ and $k = 1, \ldots, q$) of the problem $F_q(Q_1, \ldots, Q_q)|p_j = 1|C_{\max}$ such that C^*_{j1}, $j = 1, \ldots, n$, is optimal to the problem $Q_1|p_j = 1|C_{\max}$.*

Proof. This follows directly from the minimality property of solutions derived by the ECT procedure for the single-stage problem. This property asserts that no reduction can be obtained in the completion time of the jth job in the sequence, where $j = 1, \ldots, n$. □

Lemma 4. *In any optimal solution (C^*_{jk}, $j = 1, \ldots, n$ and $k = 1, \ldots, q$) of the problem $F_q(Q_1, \ldots, Q_q)$ $|p_j = 1|C_{\max}$ the schedule of the last stage (C^*_{jq} for $j = 1, \ldots, n$) is an optimal solution to the single-stage problem $Q_q|r_{jq}, p_j = 1|C_{\max}$ with release times $r_{jq} = C^*_{j,q-1}$, $j = 1, \ldots, n$.*

Proof. Regardless of how all the previous stages are scheduled, the schedule of the last stage, q, has to meet the minimum $C_{\max}$ criterion. The only constraint is the job release times, r_{jq}, which are given by the jobs' completion times at the preceding stage, $C_{j,q-1}$. □

The following algorithm, referred to as *2-Stage*, may be used to solve $F_2(Q_1, Q_2)|p_j = 1|C_{\max}$.

Algorithm 2-stage

1.0 Use the ECT rule to solve the first stage as $Q_1|p_j = 1|C_{\max}$. The procedure returns the completion times $C_{11}, \ldots, C_{n1}$.
2.0 Use the job completion times of the first stage as release times of the jobs for the second stage (i.e., $r_{j2} = C_{j1}, j = 1, \ldots, n$). Apply the LST rule to solve $Q_2|r_{j2}, p_j = 1|C_{\max}$. The procedure returns the job to machine assignments.
3.0 Schedule the jobs forward to compute the completion times at the second stage, C_{j2}, $j = 1, \ldots, n$, taken into consideration their release times, r_{j2}, and their machine assignments from Step 2. The optimal makespan, $C^{\mathrm{o}}_{\max}$, is the largest completion time of any job at the second stage.

Algorithm 2-stage is of complexity $O(n \log m + n \log n)$ since the single-stage problem solved in the first step is $O(n \log m)$ and the single-stage problem solved in the second step is $O(n \log n)$ (Dessouky et al., 1990). The complexity of the 2-stage problem is summarized by the following theorem.

Theorem 1. *$F_2(Q_1, Q_2)|p_j = 1|C_{\max}$ may be solved in polynomial-time.*

Proof. The validity of Algorithm 2-Stage follows from Lemmas 3 and 4. □

4. The three-stage problem

We first show that the 3-stage problem is NP-hard and then present an algorithm that optimally solves $F_3(Q_1, Q_2, Q_3)|p_j = 1|C_{\max}$ making use of the (partial) optimality conditions stated in Lemmas 3 and 4. We then give a specialization of the 3-stage problem which may be solved in polynomial-time.

Theorem 2. *Problem $F_3(Q_1, Q_2, Q_3)|p_j = 1|C_{\max}$ is NP-hard.*

Proof. We show that the 3-stage problem is NP-hard by showing that it is a generalization of the Numerical 3-Dimensional Matching (N3DM) Problem which is NP-complete in the strong sense (Garey and Johnson, 1979, pp. 224).

Instance: Disjoint sets W, X, and Y, each containing m elements, a size $s(a) \in Z^+$ for each element $a \in W \cup X \cup Y$, and a bound $B \in Z^+$.

Question: Can $W \cup X \cup Y$ be partitioned into m disjoint sets $A_1, A_2, \ldots, A_m$ such that each A_i contains exactly one element from each of W, X, and Y and such that for $1 \leqslant i \leqslant m, \sum_{a \in A_i} s(a) = B$?

We will assume that $\sum_{a \in W \cup X \cup Y} s(a) = mB$. The problem is still NP-complete with this assumption. Consider the general N3DM problem. We use its data to specialize $F_3(Q_1, Q_2, Q_3)|p_j = 1|C_{\max}$ in the following way. Let the members of the sets W, X, and Y represent the machines at stage 1, 2, and 3, respectively. Suppose $K = \sum_{a \in W \cup X \cup Y} s(a) + 1$ and $n = m$. For $a \in W$, let $p_{a,1} := s(a) + K$. Similarly for $a \in X$, let $p_{a,2} := s(a) + K$ and for $a \in Y$, let $p_{a,3} := s(a) + K$.

We show that the answer to $F_3(Q_1, Q_2, Q_3)|p_j = 1|C_{\max}$ is exactly $B + 3K$ if and only if N3DM has an answer in affirmative. With the above data, in the optimal schedule, no two jobs will be scheduled on the same machine. The proof is by contradiction. Suppose that in a feasible schedule, jobs k and j use the same machine at the first stage and k precedes j. The 1-stage completion time of job j would be at least $2K$. Hence, the 3-stage completion time of job j cannot be less than $4K$. But we can easily improve this schedule by not using any machine twice. Just schedule job number k on the kth machine at each stage. This way the maximum completion time of any job does not exceed $\sum_{i=1}^{3} \max_{a \in S_i} p(a)$ which is strictly less than $4K$. We can similarly handle the other two cases when some machine at stage 2 or 3 is being used more than once. Therefore, any optimal schedule to the minimum makespan problem gives us a 3-dimensional matching.

Since in any optimal schedule all machines are used exactly once, the average completion time of a job is $B + 3K$. Hence, the optimal makespan is bounded below by $B + 3K$. Moreover, it is equal to $B + 3K$ if and only if the completion time of every job in an optimal schedule is exactly $B + 3K$ which immediately translates to a 3-dimensional matching with the desired properties. This result is equivalent to showing that problem $F_3(Q_1, Q_2, Q_3)|p_j = 1|C_{\max}$ is NP-hard. □

Making use of Lemmas 3 and 4, the 3-stage problem may be decomposed into four main steps with the first three steps each solving a single-stage problem. We first state the algorithm and then prove that it provides an optimal solution to $F_3(Q_1, Q_2, Q_3)|p_j = 1|C_{\max}$.

Algorithm 3-stage

1.0 Use the ECT rule to solve the first stage as $Q_1|p_j = 1|C_{\max}$. The procedure returns the completion times $C_{11}, \ldots, C_{n1}$.

2.0 Assume an arbitrary due date d by which all jobs must be completed by in stage 3. Apply the LST rule at the third stage. The procedure returns an ordered set of latest start times, $(S_{13}, \ldots, S_{n3})$.

3.0 For the second stage, use the job completion times of the first stage as release times of the jobs for the second stage (i.e., $r_{j2} = C_{j1}, j = 1, \ldots, n$) and the latest start times of the third stage as *assignable* due dates for the second stage (i.e., $d_{j2} = S_{j3}, j = 1, \ldots, n$). Assignable due dates are a set of designated points in time which are arbitrarily assigned by the scheduler to the set of jobs rather than given as part of each job's input parameters. Let the assignment be given by the permutation $\pi = (\pi(1), \ldots, \pi(n))$ where $\pi(j)$ is the position of job J_j among the due dates. The due dates are assigned to the given release times in such a manner as to minimize the maximum lateness, $L_{\max}$, where

$$L_{\max} = \max_{j=1,\ldots,n} (C_{j2} - d_{\pi(j),2}).$$

We denote the second stage problem as $Q_2|r_{j2}, p_j = 1, d_{j2}\text{ assign}|L_{\max}$.

4.0 Schedule the jobs forward to compute the completion times at the third stage, C_{j3}, $j = 1, \dots, n$, taken into consideration their release times, r_{j3}, from Step 3 and their machine assignments from Step 2. The optimal makespan, $C^{o}_{\max}$, is the largest completion time of any job at the third stage.

Theorem 3. *Algorithm 3-Stage optimally solves* $F_3(Q_1, Q_2, Q_3)|p_j = 1|C_{\max}$.

Proof. Step 1 follows from Lemma 3, as applied to stage 1. Step 2 together with Step 4 is an application of Lemma 4 to stage 3. In Step 2, the latest start times of the jth job to start at the third stage, S_{j3}, $j = 1, \dots, n$, are obtained. In Step 4, the values of r_{j3}, $j = 1, \dots, n$, are matched in non-decreasing order with $S_{\pi(j),3}$. The jobs are then scheduled forward with consideration of their release times, r_{j3}, to determine their completion times, C_{j3}, $j = 1, \dots, n$, and $C_{\max} = \max_{j=1,\dots,n} C_{j3}$. To determine r_{j3}, define $\Delta_j = r_{j3} - S_{\pi(j),3}$ and $\Delta = \max_{j=1,\dots,n} \Delta_j$. Note that $C_{\max} = d + \Delta$. Therefore, it becomes obvious that the objective for stage 2 is to find

$$\begin{aligned} \min \Delta &= \min \max_{j=1,\dots,n} (r_{j3} - S_{\pi(j),3}) \\ &= \min \max_{j=1,\dots,n} (C_{j2} - S_{\pi(j),3}). \end{aligned}$$

Since the latest start times at the third stage are effectively due dates on the second stage, then $d_{\pi(j),2} = S_{\pi(j),3}$ and $C_{j2} - S_{\pi(j),3} = C_{j2} - d_{\pi(j),2} = L_{j2}$, the lateness of the job at the second stage. Thus, minimizing $C_{\max}$ at the third stage requires minimizing $L_{\max}$ at the second stage given the release times $r_{12}, \dots, r_{n2}$ which are simply the completion times at the first stage $C_{11}, \dots, C_{n1}$. Therefore, the second-stage problem becomes $Q_2|r_{j2}, p_j = 1, d_{j2}\ \text{assign}|L_{\max}$. □

Since $F_3(Q_1, Q_2, Q_3)|p_j = 1|C_{\max}$ is NP-hard and Steps 1, 2, and 4 of *Algorithm 3-Stage* may be solved in polynomial-time, $Q|r_j, p_j = 1, d_j\ \text{assign}|L_{\max}$ must also be NP-hard. In Section 6, we present a branch and bound procedure to solve $Q|r_j, p_j = 1, d_j\ \text{assign}|L_{\max}$.

When the processing times of all the machines are equal, $Q|r_j, p_j = 1, d_j\ \text{assign}|L_{\max}$ may be solved in polynomial-time by first calculating the completion times using the ECT rule with consideration of the release times. Then, the jobs are sorted in order of non-decreasing due dates and matched with the non-decreasing completion times. This procedure is an extension of Jackson's earliest due date rule (Jackson, 1955) for minimizing the maximum lateness on a single machine. This result is summarized in the following corollary.

Corollary 1. $F_3(Q_1, Q_2, Q_3)|p_j = 1|C_{\max}$ *with the processing times of the machines in the second stage equal may be solved in polynomial-time.*

5. Heuristic solution of the 3-stage problem

We define next a polynomial approximation algorithm to solve the 3-Stage problem. We refer to this procedure as *ECT–LST* since this procedure makes use of both the ECT and LST rules. This procedure is similar to *Algorithm 3-Stage* except for Step 3, since Steps 1 and 2 guarantee optimal solutions to Stages 1 and 3 in polynomial-time, as stated in Lemmas 3 and 4. In *Procedure ECT–LST*, Step 3 of the optimal solution to $Q_2|r_{j2}, p_j = 1, d_{j2}\ \text{assign}|L_{\max}$ is approximated by first solving $Q_2|r_{j2}, p_j = 1|C_{\max}$ and assigning d_{j2} according to non-decreasing C_{j2}. This heuristic is similar to solving the first two stages of the 3-stage problem as a 2-stage problem and then solving the third stage as a single-stage problem with unequal release times using the LST rule with the completion times at the second stage used as release times for the third stage. We now develop theoretical bounds on the heuristic solution and in Section 7 we experimentally compare the effectiveness of the heuristic solution on sample problems. We show that the maximum difference between the solution derived by the heuristic procedure and the optimal solution is bounded by the maximum processing time at the second stage, independent of the number of jobs and the processing times at the first and third stages. We also show that the heuristic provides 1.75-approximation to the optimal solution. Before presenting the proofs, we first introduce some definitions.

Now, let $C_{1k}, C_{2k}, \ldots, C_{nk}$ be the completion times of the jobs and let $t_{1k}, t_{2k}, \ldots, t_{nk}$ be the ordered completion times at stage k, $k = 1, 2$, under *Procedure ECT–LST* applied to a total of n jobs. Also, let $\bar{t}_{jk}$ be the minimum makespan of the problem $Q_k|r_{lk}, p_l = 1|C_{\max}$ with a total of j jobs, where the release time of a job is its completion time at the previous stage. The value of $\bar{t}_{jk}$ represents the minimal possible value of the jth completion time at stage k, *given* the completion times of the jobs at stage $k-1$. The sequence $(\bar{t}_{1k}, \ldots, \bar{t}_{nk})$ is thus a conditional minimal vector which may not be a feasible solution. Note that $t_{n2} = \bar{t}_{n2}$.

From the minimality property at stage 1, $\bar{t}_{j1} = t_{j1}$, for all j. For stage 2, $\bar{t}_{j2} \leqslant t_{j2}$, for all j. Let δ_k be the maximum deviation of the completion time of any job from the conditional minimality for $k = 1, 2$.

$$\delta_k = \max_{j \leqslant n}(t_{jk} - \bar{t}_{jk}) \quad \text{for } k = 1, 2,$$

δ_1 is zero because of the minimality property. However, δ_2 can be positive, playing an important role in evaluating the performance of heuristic *ECT–LST* as indicated by the following proposition.

Proposition 1. *Suppose that $C^{\mathrm{h}}_{\max}$ is the makespan of the schedule provided by Procedure ECT–LST and $C^{\mathrm{o}}_{\max}$ is the optimal makespan, then $C^{\mathrm{h}}_{\max} - C^{\mathrm{o}}_{\max} \leqslant \delta_2$.*

Proof. It is clear that a sufficient condition for the optimality of $C^{\mathrm{h}}_{\max} = t_{n3}$ is that the job J_g with the largest $L_{\max} = \max_{j \leqslant n}(t_{j2} - d_{j2}) = t_{g2} - d_{g2}$, as given by Step 3 of procedure ECT–LST, have the minimum possible t_{g2}. This is true since the values of d_{j2}, $j = 1, \ldots, n$, are the largest due dates possible given by Step 2 of the procedure. A stricter condition is that *all* t_{j2} are the minimum possible, that is, $t_{j2} = \bar{t}_{j2}$, $j = 1, \ldots, n$. Thus, the maximum departure from optimality of $C^{\mathrm{h}}_{\max}$, $C^{\mathrm{h}}_{\max} - C^{\mathrm{o}}_{\max}$, is bounded by the maximum deviation of t_{j2} from $\bar{t}_{j2}$, $j = 1, \ldots, n$. □

We now have an instrument to measure the quality of the heuristic solution, i.e., how far it is from the optimal solution. If desired, we can compute the size of this deviation, δ_2, by solving n 2-Stage problems. We next establish a more absolute bound on δ_2 in terms of the processing times. In the following theorem we show that this quantity is bounded by the processing time of the slowest machine of the second stage (call it $p_2^{\max}$).

Theorem 4. $\delta_2 \leqslant p_2^{\max}$, *independent of the number of jobs.*

Proof. Without loss of generality, suppose that all jobs $J_1, \ldots, J_n$ are indexed in order of their completion times at the first stage. For any $j < n$, jobs $J_1, \ldots, J_j$ may not necessarily be the first j jobs to complete at stage 2. We show that the maximum completion time of the jobs $J_1, \ldots, J_j$ at stage 2 (call it t'), does not exceed the quantity $\bar{t}_{j2}$ by more than $p_2^{\max}$. Since $t_{j2} \leqslant t'$, the result will follow.

The second stage release times of the first j jobs are the same independent of the total number of jobs to be scheduled. The only difference arises while performing the LST procedure on the second stage. When there are only j jobs to schedule, we start the LST rule with all the second stage machines having the same off time. On the other hand, when there are $n > j$ jobs to schedule, the off times of all the machines may not be identical when we reach the jth job. We show that $t' - \bar{t}_{j2}$ is no more than the difference between the machines' off times, which is bounded above by $p_2^{\max}$ (Property 8).

While scheduling job j, suppose the off time of machine M_i is f'_i. Note that the number of unassigned jobs left at this iteration is j (recall that in the LST rule, jobs are scheduled in reverse order). According to Property 9, assuming that $\min_i = 1, \ldots, m_2 f'_i = \bar{t}_{j2}$ does not affect the optimal assignment of these remaining jobs to the machines. With j jobs, any feasible solution to the problem $Q|f_i = \bar{t}_{j2}, r_{l2}, p_l = 1|C_{\max}$ is also feasible for the problem $Q|f_i = f'_i, r_{l2}, p_l = 1|C_{\max}$ where $f'_i \geqslant \bar{t}_{j2}$ for all i, since the f_i constraints are more relaxed. Since the former problem has a feasible solution, there exists a feasible solution to the latter problem. Therefore, according to Lemma 2 the latest start times provided by the LST rule are optimal for the problem $Q|f_i = f'_i, r_{j2}, p_j = 1|C_{\max}$. Hence, $\max_{l=1,\ldots,j}(S_{l2} + p_{g(l),2}) = \max_{i=1,\ldots,m_2}(f'_i)$, as all the machines finish processing by $\max_{i=1,\ldots,m_2}(f'_i)$ where $M_{g(l),2}$ is the machine processing the lth job, and t' is no greater than $\max_{l=1,\dots,j}(S_{l2}+p_{g(l),2})$, as the latter represents the latest completion times. Therefore for any $j=1,\dots,n$, $t_{j2}-\bar{t}_{j2} \leqslant t'-\bar{t}_{j2} \leqslant \max_{l=1,\dots,j}(S_{l2}+p_{g(l),2}) - \min_{i=1,\dots,m_2}(f_i') = \max_{i=1,\dots,m_2}(f_i') - \min_{i=1,\dots,m_2}(f_i') \leqslant p_2^{\max}$, where the last inequality follows from Property 4. □

The above results directly imply the following corollary.

Corollary 2. *If $C^{\mathrm{h}}_{\max}$ is the makespan of the schedule provided by Procedure ECT–LST, $C^{\mathrm{o}}_{\max}$ is the optimal makespan and $p_2^{\max}$ is the processing time of the slowest machine of the second stage, then: (i) $C^{\mathrm{h}}_{\max} - C^{\mathrm{o}}_{\max} \leqslant p_2^{\max}$, and since $C^{\mathrm{o}}_{\max} \to \infty$ as $n \to \infty$, (ii) $C^{\mathrm{h}}_{\max} - C^{\mathrm{o}}_{\max}/C^{\mathrm{o}}_{\max} \to 0$ as $n \to \infty$.*

One question which remains unanswered is how good an approximation does *Procedure ECT–LST* provide, i.e., what is the upper bound on the ratio $C^{\mathrm{h}}_{\max}/C^{\mathrm{o}}_{\max}$ for any n. According to the above corollary, this ratio goes to unity for large n. We would like to find out how large can this ratio be for any n. The following theorem proves that the heuristic provides a 1.75-approximation.

Theorem 5. *Procedure ECT–LST is a 1.75-approximation algorithm for the 3-stage problem.*

Proof. We wish to find an upper bound on the quantity $C^{\mathrm{h}}_{\max}/C^{\mathrm{o}}_{\max}$. Suppose that there are a total of n jobs. Then,

$$\frac{C^{\mathrm{h}}_{\max}}{C^{\mathrm{o}}_{\max}} \leqslant \frac{C^{\mathrm{o}}_{\max}+\delta_2}{C^{\mathrm{o}}_{\max}} \leqslant \frac{t_{n2}+\delta_2}{t_{n2}} = 1+\frac{\delta_2}{t_{n2}}.$$

We prove that $\delta_2/t_{n2} \leqslant 3/4$. If $t_{n2} \geqslant 4/3p_2^{\max}$, then the inequality follows because $\delta_2 \leqslant p_2^{\max}$ according to Theorem 4. Therefore, we assume that $t_{n2} < 4/3p_2^{\max}$ which immediately implies that the machine with processing time $p_2^{\max}$ is used at most once at the second stage. In fact, we can assume that it is used exactly once because if it is not used at all, then it can be removed from the problem. Since the machine with processing time $p_2^{\max}$ is used once,

$$\sum_{i=1}^{m_2} \left\lfloor \frac{p_2^{\max}}{p_{i2}} \right\rfloor \leqslant n.$$

Proof is by contradiction. Suppose that $\delta_2 > 3/4t_{n2}$. Specifically, let $t_{j2}-\bar{t}_{j2} > 3/4t_{n2}$ for some j. Therefore, the completion time of the jth job is at least $3/4t_{n2}$. Consider the second stage machine with processing time p_{i2}. The number of jobs that this machine processes after t_{j2} is at most $\lfloor (t_{n2}-t_{j2})/p_{i2} \rfloor + 1$, while the total number of jobs that this machine processes is at least $\lfloor p_2^{\max}/p_{i2} \rfloor$. The additional one unit represen's the job which may start before t_{j2} but finishes after t_{j2}. Define $n'(i) := \max(0, \lfloor p_2^{\max}/p_{i2} \rfloor - \lfloor (t_{n2}-t_{j2})/p_{i2} \rfloor - 1)$. At least $n'(i)$ jobs are completed before t_{j2} on the machine with processing time p_{i2}. Since $\sum_{i=1}^{m_2} n'(i) \leqslant j$, some machine with processing time p_{i2} will be used at least $\max(1, n'(i))$ times when the total number of jobs to be scheduled is j instead of n. Refer to this machine as $M_{i'}$. The second stage makespan is t_{n2}.

$$\begin{aligned}
t_{n2} &\geqslant \bar{t}_{j2} + \delta_2 + t_{n2} - t_{j2}\\
&\geqslant p_{i'2}\max(1, n'(i')) + \delta_2 + t_{n2} - t_{j2}\\
&> p_{i'2}\max(1, n'(i')) + 3/4t_{n2} + p_{i'2}\left\lfloor \frac{t_{n2}-t_{j2}}{p_{i'2}} \right\rfloor\\
&= 3/4t_{n2} + p_{i'2}\max\left(\left\lfloor \frac{t_{n2}-t_{j2}}{p_{i'2}} \right\rfloor + 1,\right.\\
&\qquad \left.\left\lfloor \frac{p_2^{\max}}{p_{i'2}} \right\rfloor - 1\right)\\
&\geqslant 3/4t_{n2} + p_{i'2}\max\left(1, \left\lfloor \frac{p_2^{\max}}{p_{i'2}} \right\rfloor - 1\right)\\
&> 3/4t_{n2} + 1/3p_2^{\max}\\
&> 3/4t_{n2} + 1/3(3/4t_{n2}) = t_{n2}.
\end{aligned}$$

The second strict inequality comes from the fact that for any real number $f \geqslant 1$, $(1/f)\max(1, \lfloor f \rfloor - 1) > 1/3$. Hence, the above calculations conclude that $t_{n2} > t_{n2}$, a contradiction. Hence, $\delta_2 \leqslant 3/4t_{n2}$. □

6. Branch and bound procedure

The only computationally difficult step of *Algorithm 3-Stage* is Step 3 which solves $Q|r_j, p_j = 1, d_j$ assign$|L_{\max}$. In this section, we present a branch and bound procedure to solve $Q|r_j, p_j = 1, d_j$ assign$|L_{\max}$. We note that the single machine sequencing problem to minimize the maximum lateness with *non-identical* jobs given release times and designated due dates is also NP-hard (Lenstra, 1977), and there is a significant amount of work in solving this problem based on branch and bound procedures (e.g., McMahon and Florian, 1975; Larson et al., 1985). However, we are not aware of any prior work on $Q|r_j,p_j=1,d_j$ assign$|L_{\max}$. Besides the importance of solving $Q|r_j,p_j=1,d_j$ assign$|L_{\max}$ for the 3-stage problem, this problem is interesting in its own right as the assignable due dates can be viewed as a set of designated points in time where deliveries must be made and the assignment to the jobs must be made by the scheduler rather than given as part of each job's input parameters. This freedom of assignment provides the scheduler with a greater latitude in delivering identical products against promised due dates to minimize a function of the lateness of the jobs.

We now present a lower bound for $Q|r_j,p_j=1,d_j$ assign$|L_{\max}$. Let σ be a partial schedule of jobs on the machines, R be the set of jobs not in σ, and $r_{\min}=\min_{J_j\in R} r_j$. Define the lower bound given the partial schedule σ as $\underline{L}_{\max}(\sigma)$. Set the availability time a_i of machine $M_i(i=1,\ldots,m)$ to the completion time of the last job scheduled on M_i in σ. Given a partial schedule σ of jobs on the machines, the following procedure develops a complete schedule.

```
1.0 Set $r_j = r_{\min}, J_j \in R$.
2.0 Solve the scheduling problem of jobs in $R$ as the equal release time problem of $Q|a_i, r_j = r_{\min}, p_j = 1|C_{\max}$ using the ECT rule.
3.0 Assign $d_j$ to $J_j$, $j = 1,\ldots,n$, according to a non-decreasing order of $C_j$ in the complete sequence, and compute $L_j = C_j - d_j$, $j = 1,\ldots,n$, and $\underline{L}_{\max}(\sigma) = \max_{j=1,\ldots,n} L_j$.
```

Let $L^*_{\max}(\sigma)$ be the value of $L_{\max}$ given by the optimal solution of $Q|a_i,r_j,p_j=1,d_j$ assign$|L_{\max}$ given σ.

Proposition 2. $\underline{L}_{\max}(\sigma) \leqslant L^*_{\max}(\sigma)$.

Proof. From Lemma 1 the solution of $Q|a_i,p_j=1|C_{\max}$ given σ is optimal and has the minimality property. Thus, $\underline{L}_{\max}(\sigma)$ is optimal for the problem with $r_j=r_{\min}$, $J_j\in R$. Since this problem is a relaxation of the same problem with original r_j, $J_j\in R$, then $\underline{L}_{\max}(\sigma)\leqslant L^*_{\max}(\sigma)$. □

The solution method follows a branch and bound procedure, with each node in the branching tree representing a partial schedule $\sigma=(\sigma_1,\ldots,\sigma_m)$, where σ_i $(i=1,\ldots,m)$ is the partial schedule on machine M_i. The root of the tree represents $\sigma=\emptyset$. A node at level l in the branching tree represents the allocation of jobs $J_1,\ldots,J_l$. Different nodes at this level represent allocations to different machines. Branching from a node represents the allocation of job $J_i\in R$ at the end of some σ_i, $i=1,\ldots,m$. Property 5 implies that the choice of job J_i from R in a non-decreasing order of r_j will not preclude reaching an optimum solution. Hence, we index jobs in R in a non-decreasing order of r_j. At level l, $R=(J_{l+1},\ldots,J_n)$. Given a node representing σ at level l with machines availability times a_i, $i=1,\ldots,m$, the following procedure computes the lower bound on the branch node of machine M_i.

```
1.0 Set $r'_j = r_l, j = l+1,\ldots,n$.
2.0 Set $\sigma_y = \sigma_i$, $y \neq i$, and $\sigma_i = \sigma_i \cup J_{l+1}$. Update $a_i$ to the completion time of Job $J_{l+1}$. Refer to this allocation as $\sigma^i$.
3.0 Compute the lower bound $\underline{L}_{\max}(\sigma^i)$ by solving $Q|a_i, r_j = r_l, p_j = 1|C_{\max}$ optimally using the ECT rule.
```

A list of open nodes is maintained and a node is closed if all its branches have been generated or its lower bound exceeds an existing feasible solution. Note that the maximum number of nodes that the branch and bound procedure will evaluate is $O(m^n)$.

The initial feasible solution (upper bound) for the branch and bound procedure is the solution generated by *Procedure ECT–LST*. We now present a sufficient condition to test whether the existing feasible solution is optimal in the branch and bound procedure and a pruning strategy to drop nodes which cannot lead to optimal solutions.

Proposition 3. *Given a feasible schedule in which the third stage job to machine assignments are done* *according to the LST rule (as per Step 2.0 of Algorithm 3-Stage), if there exists j such that the completion time of the job which is jth to finish at the second stage equals both the jth second stage minimum completion time and the jth third stage latest start time, then the schedule is optimal.*

Proof. The optimal third stage job to machine assignments are independent of the second stage schedule. Therefore, we can determine which third stage machine should process the jth job to complete at the second stage independent of the second stage schedule. Call this machine $M(j)$.

If $t_{j2} = S_{j2}$, it implies that machine $M(j)$ is the bottleneck machine and the off-time of $M(j)$ is equal to the makespan. The only way to reduce the makespan is to reduce the latest start time, S_{j2}. However, $t_{j2} = \bar{t}_{j2}$ according to the assumption, implying that t_{j2} cannot be reduced any further. Therefore, the current schedule is optimal. □

The basic pruning strategy is the traditional strategy based upon lower bounds used in branch and bound procedures. The following self-evident proposition provides a basis for another pruning strategy to drop offspring nodes before lower bounds are computed.

Proposition 4. *There exists an optimal schedule such that at any time during the second stage an unscheduled job with the earliest release time is not scheduled on any machine which becomes available after the job's completion time on any other machine.*

The above proposition dictates that at every node, we need to branch on only those machines which become available before the earliest completion time of the job being considered.

7. Numerical experiments

We experimentally compared the effectiveness of solutions generated by *Procedure ECT–LST* (heuristic solution) with the optimal solutions generated by *Algorithm 3-Stage*. Since the number of nodes to evaluate in the branch and bound procedure is $O(m^n)$, we tested the effectiveness of the heuristic on different combinations of the number of machines at the second stage and the number of jobs. The values tested for the number of jobs were 5, 10, 25, 50, and 100 and for the number of machines at the second stage were 2, 5, 10, and 15. The processing time of each machine at each stage is sampled from a uniform random number from 1 to 100. We set the number of machines at the first and third stages equal to the number of machines at the second stage because if there is a clearly defined bottleneck stage, the 3-stage problem essentially reduces to a 2-stage problem. Under this situation, the heuristic performs extraordinarily well. Furthermore, having a clear bottleneck stage is a situation which industry tries to avoid. Since it is desirable to test the heuristic on reasonably hard problems, we restrict our experiments to the case where there is no apparent bottleneck stage. This case is when the number of machines at each stage are equal. We have earlier proved that this special case is also NP-hard.

A depth-first branching strategy was used for the branch and bound procedure with the offspring nodes generated in order of the machine's processing time. This branching strategy was used because of its minimal storage requirements and it consistently outperformed other branching strategies (e.g., breadth-first and smallest lower bound). We stopped the branch and bound procedure after 100,000 nodes were evaluated so that many experiments could be performed. The run-time of evaluating one node in the branch and bound procedure on a SUN SPARCsystem 600 was on the average 0.0107 CPU second.

For each number of machines at the second stage and number of jobs combination 30 random samples were generated. Table 1 shows the results of the experiments. This table lists the number of times an optimal solution was found out of the 30 experiments, the number of times the heuristic procedure gave the optimal solution out of the 30 experiments, the average number of nodes evaluated in the branch and bound procedure to obtain an optimal solution, the average ratio of the heuristic solution over the optimal solution (i.e., $C^{\mathrm{h}}_{\max}/C^{\mathrm{o}}_{\max}$) if the latter is found, and the average ratio of the absolute bound over the heuristic

Table 1
Experimental results

Number of machines	Number of jobs	Optimal found [a]	Heuristic optimal [b]	Average node [c]	C^{h}_{max}/C^{o}_{max} [d]	$C^{h}_{max} + \delta_2 / C^{h}_{max}$ [e]	$C^{h}_{max} + \delta_2 / C^{h}_{max}$ [f]
2	5	30	27	0.6	1.0074	1.0431	–
	10	30	21	2.6	1.0057	1.0358	–
	25	29	23	3338.8	1.0028	1.0196	1.0029
	50	30	24	12.0	1.0018	1.0117	–
	100	30	23	13.8	1.0010	1.0069	–
5	5	30	14	30.1	1.0157	1.0719	–
	10	29	14	3360.2	1.0275	1.0898	1.1340
	25	29	13	3406.3	1.0121	1.0608	1.0586
	50	29	13	3451.9	1.0066	1.0367	1.0417
	100	29	12	3478.0	1.0042	1.0204	1.0258
10	5	30	19	17.4	1.0180	1.0720	–
	10	30	11	711.7	1.0302	1.0994	–
	25	27	12	12650.2	1.0281	1.0938	1.1386
	50	25	7	22395.1	1.0206	1.0741	1.0930
	100	26	9	15876.6	1.0115	1.0487	1.0586
15	5	30	17	21.7	1.0186	1.0755	–
	10	30	11	253.0	1.0330	1.1040	–
	25	24	6	29202.5	1.0386	1.1022	1.1502
	50	18	3	43438.4	1.0221	1.0899	1.1192
	100	16	4	51392.1	1.0153	1.0550	1.0759

[a] Number of times the branch and bound procedure found an optimal solution.
[b] Number of times the heuristic solution procedure obtained an optimal solution.
[c] The average number of nodes evaluated in the branch and bound procedure per run.
[d] The average ratio of the heuristic solution over the optimal solution.
[e] The average ratio of the absolute bound over the heuristic solution for the runs in which an optimal solution was found.
[f] The average ratio of the absolute bound over the heuristic solution for the runs in which an optimal solution was not found.

solution (i.e., $(C^{h}_{max} + \delta_2)/C^{h}_{max}$) for runs in which an optimal solution was not obtained and runs in which it was obtained. The last two columns represent how far the heuristic deviates from optimality if the optimal solution is not known.

An example of an experiment in which optimal solutions were not obtained is the experiment with 2 machines and 25 jobs. The optimal solution was not found after evaluating 100,000 nodes in 1 out of the 30 runs. The case which shows an average number of nodes evaluated less than one indicates that the sufficient condition for optimality as stated in Proposition 3 was met for some of the initial feasible solutions (heuristic solution). As Table 1 shows, the results of these experiments indicate that the heuristic solution provides a good approximation. The overall average ratio of heuristic solution over the optimal solution was 1.0156 and in 51.4% of the experiments the heuristic solution procedure gave the optimal solution. The overall average absolute bound $(C^{h}_{max} + \delta_2)$ over the heuristic solution was 1.0595 which is considerably less than the theoretical bound of 1.75. There does not appear to be much of a difference in this bound between experiments in which optimal solutions were found and the experiments in which optimal solutions were not found.

8. Conclusions and directions for future research

We have shown that the 2-stage scheduling problem with identical jobs and uniform parallel machines may be solved in polynomial-time, and the 3-stage problem is NP-hard. For $F3(Q_1, Q_2, Q_3)|p_j = 1|C_{max}$, *Algorithm 3-stage* optimally solves the problem by solving three single-stage problems. The only computationally difficult step is the procedure for the second stage which must solve $Q|r_j, p_j = 1, d_j\,\text{assign}|L_{\max}$. A polynomial heuristic procedure is presented to approximate the 3-stage problem. We showed that the maximum deviation between the solution derived by the heuristic procedure and the optimal solution is bounded by the maximum difference between the jth completion time at the second stage and the jth minimal completion time at the second stage for $j = 1, \ldots, n$ which is bounded by the maximum processing time at the second stage, independent of the number of jobs and the processing times at the first and third stage. We also showed that the heuristic is a 1.75-approximation algorithm for the 3-stage problem.

The heuristic procedure *ECT–LST* may be easily generalized to consider an arbitrary number of stages, q. In this case, the first stage is solved using the ECT rule and all other stages are solved using the LST rule with the release times at a given stage set to the completion times of the previous stage. In a similar manner to the 3-stage problem, it can be shown that this heuristic applied to q-stages will have a maximum deviation from the optimal solution of $\sum_{k=2}^{q-1} p_k^{\max}$ where $p_k^{\max}$ is the maximum processing time of any machine at stage k (Dessouky et al., 1996). Future research can focus on developing relative bounds for this heuristic and an optimal solution procedure for more than 3 stages.

Acknowledgements

The authors are indebted to S.L. van de Velde, whose work on scheduling a garment manufacturer's operations provided the definition of the problem addressed in this paper.

References

Cheng, T.C.E., Sin, C.C.S., 1990. A state-of-the-art review of parallel-machine scheduling research. Eur. J. Oper. Res. 47, 271–290.

Dessouky, M.I., Lageweg, B.J., Lenstra, J.K., van de Velde, S.L., 1990. Scheduling identical jobs on uniform parallel machines. Statistica Neerlandica 44, 115–123.

Dessouky, M.M., Dessouky, M.I., Verma, S., 1996. A note on flowshops with a large number of stages, identical jobs, and uniform parallel machines. Technical Paper # 1996–04, University of Southern California, Los Angeles, CA.

Garey, M.R., Johnson, D.S., 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, New York.

Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., 1979. Optimization and approximation in determining sequencing and scheduling: A survey. Ann. Discrete Math. 5, 287–326.

Guinet, A.G.P., Solomon, M.M., 1996. Scheduling hybrid flowshops to minimize maximum tardiness or maximum completion-time. International Journal of Production Research 34, 1643–1654.

Gupta, J.N.D., 1988. Two stage hybrid flowshop scheduling problem. Journal of Operational Research Society 39, 359–364.

Gupta, J.N.D., Tunc, E.A., 1994. Scheduling a 2-stage hybrid flowship with separable setup and removal times. Eur. J. Oper. Res. 77, 415–428.

Jackson, J.R., 1955. Scheduling a production line to minimize maximum lateness. Research Report #43, Management Science Research Project, UCLA, Los Angeles, CA.

Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., 1982. Recent developments in deterministic sequencing and scheduling. In: Dempster, M.A.H., Lenstra, J.K., Rinnooy Kan, A.H.G. (Eds.), Deterministic and Stochastic Scheduling. Reidel, Dordrecht, 35–73.

Larson, R.E., Dessouky, M.I., DeVor, R.E., 1985. A forward-backward procedure for the single machine problem to minimize maximum lateness. IIE Transactions 17, 252–260.

Lee, C.Y., Vairaktarakis, G.L., 1994. Minimizing makespan in hybrid flowshops. Oper. Res. Lett. 16, 149–158.

Lenstra, J.K., 1977. Sequencing by enumerative methods. Mathematical Centre Tracts, No. 69, Mathematics Centrum, Amsterdam, The Netherlands.

McMahon, G., Florian, M., 1975. On scheduling with ready times and due dates to minimize maximum lateness. Oper. Res. 23, 475–482.