Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

J.A. Hoogeveen, S.L. van de Velde

Scheduling around a small common due date

Department of Operations Research, Statistics, and System Theory Report BS-R8914 July

The Centre for Mathematics and Computer Science is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Research (N.W.O.).

Scheduling Around a Small Common Due Date

J.A. Hoogeveen
S.L. van de Velde
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam
The Netherlands

A set of n jobs has to be scheduled on a single machine which can handle only one job at a time. Each job requires a given positive uninterrupted processing time and has a positive weight. The problem is to find a schedule that minimizes the sum of weighted deviations of the job completion times from a given common due date d, which is smaller than the sum of the processing times. We prove that this problem is $\mathcal{NP}$-hard even if all job weights are equal. In addition, we present a pseudopolynomial algorithm that requires $O(n^2 d)$ time and $O(nd)$ space.

1980 Mathematics Subject Classification (1985 Revision): 90B35.
Key Words & Phrases: single machine scheduling, $\mathcal{NP}$-hardness, pseudopolynomial algorithm.

1. Introduction

Suppose n independent jobs have to be scheduled on a single machine which can handle only one job at a time. The machine is assumed to be continuously available from time 0 onwards. Job J_i ($i = 1,...,n$) has a given positive uninterrupted processing time p_i and a weight w_i, and should ideally be completed at a given due date d, which is common to all jobs. Without loss of generality, we assume that the processing times and the due date are integral. A *schedule* defines for each job J_i a completion time C_i such that the jobs do not overlap in their execution. We consider the problem of finding a schedule S, that minimizes the weighted sum of the deviations of the completion times from the common due date:

$$f(S) = \sum_{i=1}^{n} w_i \,|\, C_i - d \,|.$$

There are two notable results in case $d \geqslant \Sigma p_i$. If $w_i = 1$ for all J_i, Kanet (1981) has given an $O(n \log n)$ time algorithm to find the optimal schedule. Hall and Posner (1989) have shown that the problem with arbitrary weights is $\mathcal{NP}$-hard.

In contrast, we focus our attention to the case $d < \Sigma p_i$. In Section 2 we prove some properties of an optimal schedule. In Section 3 we establish $\mathcal{NP}$-hardness of this problem, even in case $w_i = 1$ for each J_i. We present a pseudopolynomial algorithm in Section 4, which requires $O(n^2 d)$ time and $O(nd)$ space. In Section 5 we present some well-solvable cases.

We note that the $\mathcal{NP}$-hardness result was independently obtained by Hall, Kubiak and Sethi (1989); their proof is slightly more complicated. They also give a pseudopolynomial algorithm for the case of equal job weights.

Report BS-R8914
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

2. Basic concepts

It is straightforward to verify that no optimal solution has any idle time between the execution of jobs. In case there were idle time, the scheduling cost could be reduced by closing the gap. The next two theorems further characterize any optimal solution.

THEOREM 1. *In any optimal schedule S, the jobs J_i that are completed before or at the common due date d are scheduled in order of nondecreasing values of w_i / p_i, and the jobs that are started at or after d are scheduled in order of nonincreasing values of w_i / p_i.*

PROOF. This follows immediately from Smith's ratio rule (Smith, 1956). □

THEOREM 2. *In each optimal schedule S, either the first job starts at time 0 or the due date d coincides with the start time or completion time of the job with the largest ratio w_i / p_i.*

PROOF. For a given schedule S, let $B(S)$ denote the set of jobs that are completed before or at the common due date and $A(S)$ the set of jobs completed after the due date. Define $\Delta = \Sigma_{J_i \in B(S)} w_i - \Sigma_{J_i \in A(S)} w_i$. We consider the cases in which $\Delta < 0$ and $\Delta \geqslant 0$ separately.

Suppose first $\Delta < 0$. If S starts at time $T > 0$, determine $t = \min\{T, \min_{J_i \in A(S)} C_i - d\}$. If the entire schedule is put t time units earlier, then the reduction in cost equals $-t\Delta > 0$. In the new situation either schedule S starts at time $T = 0$ or one job has moved from $A(S)$ to $B(S)$. If still $T > 0$ and $\Delta < 0$, we repeat the procedure until we arrive at a situation in which $T = 0$ or $\Delta \geqslant 0$, and no further improvement is possible. The latter case implies that the due date coincides with the completion time of one job and the start time of another. Because of Theorem 1, one of these jobs must be the job with the largest ratio w_i / p_i.

On the other hand, in the case of $\Delta \geqslant 0$, reverse arguments can be applied to show that the due date coincides with the completion or start time of the job with the largest ratio w_i / p_i. □

Note that Theorem 1 does not impose any restrictions on a job that is started before and completed after the due date. Consider the following instance with $n = 3, p_1 = 8, p_2 = 10, p_3 = 4, w_1 = 5, w_2 = 7, w_3 = 3$, and $d = 15$. The optimal solution is shown in Figure 1 and demonstrates that such a job can exist, and that it can even have the smallest ratio w_i / p_i.

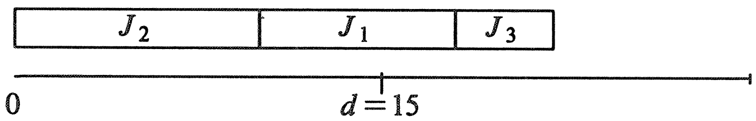

FIGURE 1

3. Scheduling around a small common due date is $\mathcal{NP}$-hard

In this section we prove that this problem is $\mathcal{NP}$-hard even if $w_i = 1$ for each job J_i, by showing that the corresponding decision problem is $\mathcal{NP}$-complete. The reduction is from Even-Odd Partition.

EVEN-ODD PARTITION (Garey et al., 1988): Given a set of $2n$ positive integers $B = \{b_1, \ldots, b_{2n}\}$ such that $b_i > b_{i+1}$ for each $i = 1,...,2n-1$, is there a partition of B into two subsets B_1 and B_2 such that $\Sigma_{b_i \in B_1} b_i = \Sigma_{b_i \in B_2} b_i = A$ and such that B_1 contains exactly one of $\{b_{2i-1}, b_{2i}\}$ for each $i = 1,...,n$?

We start by describing a reduction from the Even-Odd Partition problem to the small common due date problem with $w_i = 1$ for all J_i. Let $B = \{b_1,...,b_{2n}\}$ be an arbitrary instance of the Even-Odd Partition problem, with $A = \Sigma b_i / 2$. Construct the following set of jobs: $2n$ 'partition' jobs J_i with processing times $p_i = b_i + nA$ for each $i = 1,...,2n$, an additional job J_0 with $p_0 = 3(n^2+1)A$, weights $w_i = 1$ for $i = 0,...,2n$, and a common due date $d = (n^2+1)A$. In addition, we define a threshold value $y_0 = \Sigma_{i=1}^{n} [(i+1)(p_{2i-1} + p_{2i})] + d$ on the scheduling cost.

Consider a partitioning of the set of partition jobs $\{J_1, \ldots, J_{2n}\}$ into the sets $B_1 = \{J_{11}, J_{21}, \ldots, J_{n1}\}$ and $B_2 = \{J_{12}, J_{22}, \ldots, J_{n2}\}$, where $\{J_{i1}, J_{i2}\} = \{J_{2i-1}, J_{2i}\}$ for each $i = 1,...,n$.

LEMMA 1. *If the partitioning into the sets B_1 and B_2 corresponds to a solution of the Even-Odd Partition problem, then the cost of schedule S_0 constructed as shown in Figure 2 equals the threshold value y_0.*

FIGURE 2: SCHEDULE S_0

PROOF. Note that the jobs in B_1 and B_2 are scheduled as indicated in Theorem 1. The verification then only requires straightforward computations. □

We now prove that, conversely, any schedule S with $f(S) \leq y_0$ must be isomorphic to S_0, and that the subsets B_1 and B_2 must correspond to a solution of the Even-Odd Partition problem.

PROPOSITION 1. *Suppose S is a schedule with scheduling cost $f(S) \leq y_0$. Then S has the following properties.*
(1) *At most n jobs can be completed before the due date d.*
(2) *The first job must start at time 0.*
(3) *The additional job J_0 is scheduled last.*
(4) *At least $n-1$ jobs must be completed before the due date d.*

PROOF.
(1) This is due to the choice of the processing times.
(2) This follows immediately from the first property and the proof of Theorem 2.
(3) Suppose J_0 is not scheduled last. Then, because of Theorem 1, J_0 must start before the

common due date d. Since at most n jobs can be scheduled before job J_0, for at least $n+1$ jobs in S we have $C_i - d \geqslant p_0 - d = 2d$. This implies $f(S) \geqslant 2(n+1)d \geqslant (n+4)d$. However,

$$y_0 = \sum_{i=1}^{n}[(i+1)(p_{2i-1}+p_{2i})] + d < \tfrac{1}{2}(n+3)\sum_{i=1}^{2n} p_i + d = (n+4)d \leqslant f(S),$$

which contradicts the assumption.
(4) This follows immediately from the first three properties and the choice of the processing times. □

LEMMA 2. *Suppose S is an optimal schedule with $f(S) \leqslant y_0$. Then the due date d must coincide with the completion time of the n-th job in the schedule S, the schedule S must be isomorphic to the schedule S_0, and provide an affirmative answer to the Even-Odd Partition problem.*

PROOF. Assume that $s(i)$ denotes the index of the job that is scheduled on position i in schedule S. We compute the scheduling cost relative to the imaginary due date $k = p_{s(1)} + ... + p_{s(n)}$. Then we have

$$\begin{aligned}\sum_{i=0}^{2n} |C_i - k| &= \sum_{i=1}^{n}[(i-1)p_{s(i)}] + \sum_{i=n+1}^{2n}[(2n+2-i)p_{s(i)}] + 3d = \\ &\sum_{i=1}^{n}[(i+1)p_{s(i)}] + \sum_{i=n+1}^{2n}[(2n+2-i)p_{s(i)}] + 3d - 2k = \\ &\sum_{i=1}^{n}[(i+1)p_{s(i)}] + \sum_{i=1}^{n}[(i+1)p_{s(2n+1-i)}] + 3d - 2k \geqslant \\ &\sum_{i=1}^{n}[(i+1)(p_{2i-1}+p_{2i})] + 3d - 2k = y_0 + 2d - 2k.\end{aligned}$$

The true scheduling cost $f(S)$ can be written as

$$f(S) = \sum_{i=0}^{2n} |C_i - d| = \sum_{i=0}^{2n} |C_i - k| + (d-k)(card(B(S)) - card(A(S))),$$

where *card* denotes the cardinality function. Because of Proposition 1, we have only three cases to consider:

- if $d = k$, then $f(S) \geqslant y_0$,
- if $d > k$, then $card(B(S)) = n$, and therefore $f(S) \geqslant y_0 + d - k > y_0$,
- if $d < k$, then $card(B(S)) = n - 1$, and hence $f(S) \geqslant y_0 + k - d > y_0$.

This implies that if $f(S) \leqslant y_0$, then $C_{s(n)} = d$, that is, the completion time of the n-th job in S must coincide with the due date. Furthermore, $f(S) \leqslant y_0$ implies $\{J_{s(i)}, J_{s(2n+1-i)}\} = \{J_{2i-1}, J_{2i}\}$ for $i = 1,...,n$. Therefore, the schedule S is isomorphic to the schedule S_0 depicted in Figure 2. This means that the original Even-Odd Partition problem has an affirmative answer. □

THEOREM 3. *Given a set of jobs and a nonnegative integer y, the problem of deciding whether there exists a schedule S with $f(S) \leqslant y$ is $\mathcal{NP}$-complete.*

PROOF. The decision problem is clearly in $\mathcal{NP}$. For any given instance of the Even-Odd Partition problem, we construct a set of jobs as described above and set $y = y_0$. This reduction requires polynomial time. Theorem 3 now follows from Lemmas 1 and 2. □

4. A DYNAMIC PROGRAMMING ALGORITHM

Theorem 3 implies that, unless $\mathcal{P}=\mathcal{NP}$, no polynomial algorithm exists for solving the small common due date problem. We present a pseudopolynomial algorithm that requires $O(n^2d)$ time and $O(nd)$ space, for which Theorems 1 and 2 provide the basis. According to Theorem 2 we must consider two cases: one in which the job with the largest weight to processing time ratio is scheduled such that either its completion or its start time coincides with the due date, and one in which all the jobs are scheduled in the interval $[0, \Sigma p_i]$.

For the first option, we renumber the jobs according to nonincreasing weight to processing time ratios. Let $F_j(t)$ denote the optimal objective value for the first j jobs subject to the condition that the interval $[d-t,\ d+\Sigma_{i=1}^{j} p_i - t]$ is occupied by the first j jobs. Then the initialization is

$$F_j(t) = \begin{cases} 0 & \text{for } t=0, j=0, \\ \infty & \text{otherwise,} \end{cases}$$

and the recursion for $j = 1,...,n$ is given by

$$F_j(t) = \min\{F_{j-1}(t-p_j) + w_j(t-p_j),\ F_{j-1}(t) + w_j(\textstyle\sum_{i=1}^{j} p_i - t)\} \quad \text{for } 0 \leqslant t \leqslant d.$$

In the second case, all jobs are scheduled in the interval $[0, \Sigma p_i]$. In such a situation it might occur that one of the jobs is started before and yet completed after the due date (see Figure 1). To allow for this possibility, we leave one job out of the recursion, and repeat the recursion n times, once for each job. Since the cost of the schedule can now only be computed relative to the endpoints of the interval, it is assumed that the jobs have been renumbered according to nondecreasing values of w_i/p_i. Consequently, we know that the first job either starts at time 0 or finishes at time Σp_i.

Assume that J_h is the job that will be scheduled around the due date. Let $G_j^h(t)$ denote the optimal cost for the first j jobs subject to the condition that the intervals $[0,t]$ and $[\Sigma_{i=j+1}^{n} p_i + t,\ \Sigma p_i]$ are occupied by the first j jobs. The initialization is

$$G_j^h(t) = \begin{cases} 0 & \text{for } t=0, j=0, \\ \infty & \text{otherwise,} \end{cases}$$

and the recursion for $j = 1,...,n$ is

$$G_j^h(t) = \begin{cases} G_{j-1}^h(t) & \text{if } j=h, \\ G_{j-1}^h(t) + w_j(\sum_{i=j}^{n} p_i + t - d) & \text{if } d - p_j \leqslant t \leqslant d, \\ G_{j-1}^h(t-p_j) + w_j(d-t) & \text{if } \Sigma_{j+1}^{n} p_i < d-t, \\ \min\{G_{j-1}^h(t) + w_j(\sum_{i=j}^{n} p_i + t - d), G_{j-1}^h(t-p_j) + w_j(d-t)\} & \text{otherwise.} \end{cases}$$

The recursion leaves the interval $[t, t+p_h]$ idle, and it is here that we insert the job J_h and compute the resulting cost as

$$G_n^h(t) = \begin{cases} G_n^h(t) + w_h(t+p_h-d) & \text{if } d-p_h \leqslant t \leqslant d, \\ \infty & \text{otherwise.} \end{cases}$$

The optimal solution is then found as

$$f(S) = \min\{\min_{1 \leqslant h \leqslant n} \min_{d - p_h \leqslant t \leqslant d} G_n^h(t), \min_{0 \leqslant t \leqslant d} F_n(t)\},$$

by which we have established the following result.

THEOREM 4. *The dynamic programming algorithm solves the problem in $O(n^2 d)$ time and $O(nd)$ space.*

5. POLYNOMIALLY SOLVABLE CASES

5.1 *Identical jobs*

If the jobs are identical, we have $p_i = p$ for each job J_i. Since the processing times and due date are assumed to be integral, this situation is more general than the one in which all $p_i = 1$. Suppose the jobs have been renumbered according to nonincreasing weights.

If $d \geqslant p\lceil n/2 \rceil$, then it is easy to show that Emmons' matching approach (Emmons, 1987) generates an optimal schedule S by partitioning the jobs into sets $A(S) = \{J_{2i} \mid i = 1,..., \lfloor n/2 \rfloor\}$ and $B(S) = \{J_{2i-1} \mid i = 1,..., \lceil n/2 \rceil\}$, where the first job in $B(S)$ starts at time $t = d - \Sigma_{J_i \in B(S)} p_i = d - p\lceil n/2 \rceil$. In this notation, $\lfloor n/2 \rfloor$ denotes the largest integer smaller than or equal to $n/2$, and $\lceil n/2 \rceil$ denotes the smallest integer greater than or equal to $n/2$.

Conversely, if $d < p\lceil n?2 \rceil$, then there are two options: either the first job starts at time 0 or the last job in $B(S)$ is completed at time d. It is easy to see that in both cases Emmons' matching approach generates optimal schedules, and the problem is solved by choosing the better one.

5.2 *The jobs have equal weight to processing time ratios*

THEOREM 5. *In the event that $p_i = w_i$ for each job J_i, there is an optimal schedule for any value of d in which the jobs are scheduled according to nonincreasing processing times.*

PROOF. Consider two adjacent jobs that are not scheduled according to the indicated order. If both jobs are completed before or started after the common due date, then these jobs can be interchanged without affecting the cost of the schedule S. If the due date lies in the interval between the start time of the first and the completion time of the other job, then straightforward computations show that the interchange of these two jobs does not increase the cost of the schedule. □

Assume that the jobs have been renumbered according to nonincreasing processing times. Suppose r is the smallest index for which $\Sigma_{i=1}^{r} p_i \geqslant \Sigma_{i=r+1}^{n} p_i$. Theorem 5 then implies that, if $d \geqslant \Sigma_{i=1}^{r} p_i$, the problem is solved by putting $B(S) = \{J_i \mid i = 1,...,r\}$ and $A(S) = \{J_i \mid i = r+1,...,n\}$. If $d < \Sigma_{i=1}^{r} p_i$, the first job needs to start at time 0, and the jobs are processed in order of nondecreasing processing times.

REFERENCES

H. EMMONS (1987). Scheduling to a common due date on parallel common processors. *Naval Research Logistics 34*, 803-810.

M.R. GAREY, R.E. TARJAN, G.T. WILFONG (1988). One-processor scheduling with earliness and tardiness penalties. *Mathematics of Operations Research 13*, 330-348.

N.G. HALL, W. KUBIAK, S.P. SETHI (1989). *Deviation of completion times about a restrictive common due date*, Working Paper 89-19, The Ohio State University, Columbus.

N.G. HALL, M.E. POSNER (1989). *Weighted deviation of completion times about a common due date*, Working Paper 89-15, The Ohio State University, Columbus.

J.J. KANET (1981). Minimizing the average deviation of job completion times about a common due date. *Naval Research Logistics Quarterly 28*, 643-651.

W.E. SMITH (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly 3*, 59-66.