On the Complexity and Some Properties of Multi-Stage Scheduling Problems with Earliness and Tardiness Penalties

Volker Lauff and Frank Werner *
Otto-von-Guericke-Universität, Fakultät für Mathematik,
PSF 4120, 39016 Magdeburg, Germany

October 8, 2002

Abstract

In this paper, the complexity and some other properties of several multi-stage problems with a non-regular performance measure are investigated. Mainly, we deal with two-machine problems. The problems are extensions of a one-machine problem with a common due date and a non-regular optimization criterion, where the sum of absolute deviations of the completion times from the due date is to be minimized. There are two reasonable generalizations: Either, the objective function is kept the same as in the one-machine problem, where the completion time is now the completion time of the last operation of the job, or a second penalty is added to this term which takes into account the storage costs of the intermediate. For both types of problems, we study the flow shop, the job shop, and the open shop environment. In the case of intermediate storage costs, we distinguish the cases of a due date $d = 0$ (which leads to a regular optimization criterion) and a non-restrictive due date. We obtain the following results. The two-machine flow shop problem with intermediate storage costs defined in this paper is NP-hard in the strong sense for $d = 0$ as well as for a non-restrictive due date. We prove the similar results for the two-machine open shop problem. In absence of intermediate storage costs, the open shop and the job shop problem are polynomially solvable for a non-restrictive due date.

Scope and purpose

The just-in-time production philosophy has lead to a growing interest in scheduling problems considering both earliness and tardiness penalties. Most

*corresponding author, email: frank.werner@mathematik.uni-magdeburg.de

publications considering these so-called non-regular objective functions are devoted to single-stage problems. This paper describes the two main approaches to extend the objective function to multi-stage environments and investigates the computational complexity of the problems obtained. The investigations in this paper may serve as a starting point for the development of appropriate algorithms for the type of problem considered.
Keywords: scheduling, flow shop, non-regular criterion, common due date

1 Motivation and Introduction

In this paper, we will deal with a broad spectrum of objective functions and hence our results will apply to many different problems occurring in manufacturing. In principle, these objective functions include at least two of three different types of penalties, which can be motivated as follows:

1. Penalties arising from exceeding the contractually alotted delivery date are the most common (explicit contract penalties, consumer dissatisfaction).

2. A penalty for an early completion of a job is appropriate for modelling capital intensive manufacturing processes, where the costs for bounded capital are an important part of the overall costs.

3. Intermediate storage costs contribute considerably to overall costs in chemical industry, if the intermediate is not stable.

Obviously, all these costs appear in practice, but their importance varies with the problem type.

A significant part of scheduling literature is dealing with problems based exclusively on penalties of type 1. Nevertheless, the growing interest in industry in just-in-time production has created a need for mathematical models which are capable of reflecting the underlying concept. As a consequence, in addition to the common penalties (tardiness - type 1) for a completion after the negotiated due date d for all jobs, there have been introduced earliness penalties (type 2) for a completion before the due date.

One-machine problems with earliness and tardiness penalties have been studied intensively. One of the earliest papers on this topic is the work of Kanet.[1] He studied the minimization of the total deviation from a common non-restrictive due date. A due date is non-restrictive if it is not early enough to act as a constraint on the scheduling decision. We will discuss the term non-restrictive thoroughly in Section 2. Kanet provided an algorithm to solve the problem optimally for jobs of set $N = \{1, \ldots, n\}$ in $O(n \log n)$ time: Let $m = \lfloor \frac{n+1}{2} \rfloor$. A permutation $\pi = (\pi_1, \ldots, \pi_n)$ of the jobs $i = 1, \ldots, n$ is generated such that for the processing times p_i $(i = 1, \ldots, n)$ of the jobs holds $p_{\pi_1} \geq p_{\pi_n} \geq p_{\pi_2} \geq p_{\pi_{n-1}} \geq \cdots \geq p_{\pi_m}$.

We will refer to this permutation as the "Kanet-order". Then the jobs are scheduled such that there is no idle-time in-between the jobs. Thus, the schedule is completely determined by, for example, the completion time C_{π_m} of job π_m. If n is odd, the completion time $C_{\pi_m} = d$ yields an optimal schedule. For even n, all values of the interval $[d, d + p_m]$ for the completion time C_{π_m} define an optimal schedule. For the case of a restrictive due date, the problem is NP-hard in the ordinary sense.[2]

In contrast to one-stage problems, there are only a few papers dealing with multi-stage systems. There are two obvious extensions of the one-machine problem to a flow shop problem with μ machines. First, one can simply sum up the absolute deviations of the total completion times C_i of the jobs from the due date. Sarper[3] studied the problem $F2|d_i = d|\sum |C_i - d|$ (notation according to Graham et al.[4]) giving a mixed integer formulation of the problem and three heuristics. Gupta et al.[5] investigated the problem $F2|d_i = d|\sum f_i(C_i)$, where the functions $f_i(x)$ are non-increasing for $x \leq d$ and non-decreasing for $x \geq d$. In the following, we denote this monotonicity condition as Condition (M). For the special case of $f_i = f$ for all jobs, they presented an enumerative algorithm capable of solving instances with up to 20 jobs on a PC over a wide range of due date values. A second generalization of the one-machine problem is to add the storage costs of the intermediates (penalties of type 3). Denote with $s_{i\,j}$ ($c_{i\,j}$) the starting (completion) time of job i on machine M_j ($j = 1, \ldots, \mu$). A straightforward way of implementing intermediate storage costs is to simply add the term $\sum_{j=1}^{\mu-1} \sum_{i=1}^{n} (s_{i\,j+1} - c_{i\,j})$ to the objective function of the corresponding one-machine problem. Note that both types of extensions to the flow shop have their merits. The first is more convenient if the storage costs of the intermediates are negligible, whereas the second is more appropriate if both costs are comparable. Both approaches are taken into account in the present paper.

For sake of brevity, we will not give the condition $d_i = d$ explicitly in the problem notation if the common due date occurs explicitly in the objective function. Thus, we will write $F2||\sum |C_i - d| + \sum (s_{i\,2} - c_{i\,1})$ instead of $F2|d_i = d|\sum |C_i - d| + \sum (s_{i\,2} - c_{i\,1})$ from now on.

The paper is organized as follows. Section 2 shows that the models without intermediate storage costs are polynomially solvable for the case of a non-restrictive due date. Then, Section 3 gives some basic properties for the problem $F\mu||\sum |C_i - d| + \sum_{j=1}^{\mu-1} \sum_{i=1}^{n} (s_{i\,j+1} - c_{i\,j})$ and an enumerative algorithm for $F2||\sum |C_i - d| + \sum (s_{i\,2} - c_{i\,1})$. In Section 4, we prove that this problem is NP-hard in the strong sense for a restrictive and a non-restrictive due date. In Section 5, we prove the same claim for the corresponding open shop problem.

2 Constraintness and Restrictivity

In this section, we will discuss two terms which are used to classify non-regular problems.

2.1 Constraintness

For regular optimization criteria, the objective function value of a schedule cannot be reduced by inserting additional idle time, due to the monotonicity property of the objective function. An optimal schedule of the Kanet-problem, however, may have idle time before the first job is processed. Szwarc[6] distinguishes two models:

Definition 1 *Denote with t the starting time of the first job in a non-regular one-machine problem. If t is fixed, the problem is called* constrained. *If t is not fixed, the problem is called* unconstrained.

Unless mentioned explicitly, we consider *only* unconstrained problems.

2.2 Restrictivity

We will now analyze the unconstrained problems further. Kanet assumes in his article that $d \geq \sum_{i=1}^{n} p_i$. In his paper, he calls such a due date "restricted" in order to express that this due date cannot be chosen arbitrarily. Nevertheless, the term "restrictive due date" is used in contradictory meanings in the papers which appeared later. This is due to the fact that the value of the due date relative to the problem data strongly influences the validity of algorithms developed for a problem type. Whereas Kanet's algorithm is not valid for relatively small due dates, the branch and bound algorithm developed by Bagchi et al.[7] assumes that the due date is sufficiently small. In general, the value of the due date may influence the computational complexity. From another point of view, we see that it follows from Condition (M) that the optimal objective function value of a certain problem cannot increase by increasing the due date while keeping constant all other problem data.

In the literature, both aspects — computational complexity and optimal objective function value — are considered. Baker and Scudder,[8] referring to a result by Bagchi et al.,[7] strengthen Kanet's result by calling a due date d non-restrictive if

$$d \geq \Delta, \tag{1}$$

where

$$\Delta = \begin{cases} p_n + p_{n-2} + p_{n-4} + \cdots + p_2, & \text{if } n \text{ is even} \\ p_n + p_{n-2} + p_{n-4} + \cdots + p_1, & \text{if } n \text{ is odd} \end{cases}, \tag{2}$$

which is in fact the minimal due date for which a Kanet-schedule remains to be feasible. This is restrictivity rather from the point of view of complexity.

On the other hand, Webster[9] states: "If the optimal cost cannot decrease with increases in the common due date, then the problem is an unrestricted common due date problem ... The restricted common due date problem is generally much more difficult to solve."

These definitions are compatible with some problems with one machine. As an example, consider the problem $1||\sum_{i=1}^{n}|C_i - d|$. If $d < \Delta$ (which violates Equation (1)), the optimal objective function value is in general higher than the objective function value for the Kanet schedule with $d \geq \Delta$. Nevertheless, the two definitions are different for more complicated problems. The lowest possible optimal objective function value of problem $F4||\sum|C_i - d|$ is the same as the objective function value of the Kanet-schedule with the processing times on M_4 of the jobs as input data. The question whether there exists a schedule with such an objective function value for a certain due date is NP-hard in the strong sense. Consider as an example the following special case: Let the processing times on M_4 be zero for all jobs. The optimal objective function value is 0. Let $\mathrm{C}^*_{\max}$ be the optimal objective function value of $F3||\mathrm{C}_{\max}$ with the processing times on the first three machines of the jobs of the original problem as input data. Obviously, $d = \mathrm{C}^*_{\max}$ is the minimal due date for which the optimal objective function value of problem $F4||\sum|C_i - d|$ can be reached. $F3||\mathrm{C}_{\max}$ is NP-hard in the strong sense.[10] Hence, we are of the opinion that Webster's definition is not appropriate for multi-stage scheduling problems.

We propose the following definition, which provides that both aspects are preserved for multi-stage systems:

Definition 2 *For a scheduling problem with certain input data denote with F^*_d the optimal objective function value for a due date value d. Further, let $\mathfrak{S}_d$ be the set of all optimal schedules for a due date d. A due date $\tilde{d}$ is called non-restrictive if:*

$$F^*_{\tilde{d}} = \min_d\{F^*_d\} \tag{3}$$

and if there exists a schedule $\tilde{S} \in \mathfrak{S}_{\tilde{d}}$ with the property that there is no d which fulfills the two conditions

- $F^*_{\tilde{d}} = F^*_d$ *and*
- *there is a schedule $S_d \in \mathfrak{S}_d$ which can be determined with lower computational complexity than $\tilde{S}$.*

Speaking informally: A due date is non-restrictive, if and only if by increasing the due date neither the objective function value nor the complexity of the optimization problem can be reduced.

As we see from the above discussion, the decision whether some due date is non-restrictive or not may be an NP-hard problem. However, there is an sufficient criterion, which is easy to verify, for a due date to be non-restrictive: If the due date is greater than or equal to the sum of all processing times on all machines, the due date is non-restrictive.

2.3 Constraintness versus Restrictivity

In the literature, the terms constraintness and restrictivity are mixed up. One reason may be the misleading intuition that an optimal schedule for a restrictive due date necessarily starts at time 0. For instance, Bagchi et al.,[7] considering problem $1||\sum|C_i - d|$, write: "When $d < \Delta$, it should be evident that in an optimal schedule jobs are again processed continuously from $t = 0$ to $t = \sum_{i=1}^{n} p_i$; however, there is no guarantee that an optimal schedule will contain no job j with $s_j < d$ and $c_j > d$[1]." Szwarc[6] gives as a counterexample a problem with restrictive due date, where the only optimal schedules have idle time between time 0 and the starting time of the first processed job. (Nevertheless, there is always an optimal schedule for problem $1||\sum|C_i - d|$ which has either no idle time between time 0 and the starting time of the first processed job or a job completing exactly at the due date.) The definitions given above are an attempt to distinguish the terms more accurately. According to this, a problem may be either constrained or unconstrained. An unconstrained problem may have a restrictive or a non-restrictive due date. With this definition, we are able to classify the problems without intermediate storage costs. For a non-restrictive due date, the problem $J\mu||\sum|C_i - d|$ reduces to the following problem. Let P_i $(i = 1, \ldots, \mu)$ be the set of processing times on machine M_i of the jobs whose last operation has to be processed on M_i. The job shop problem reduces to μ independent $1||\sum|C_i - d|$ problems with sets P_i as input data. Each of these μ problems can be solved by Kanet's algorithm in $O(|P_i| \log |P_i|)$ time. The remaining (non-last) operations can be scheduled feasibly in an arbitrary way. Therefore, the overall complexity is $O(\mu n \log n)$. Analogously, we find that the objective function value of problem $F\mu||\sum|C_i - d|$ can be calculated in $O(n \log n)$ time and the complete scheduling problem can be solved optimally in $O(n \log n + n\mu)$ time for a non-restrictive common due date. Especially, there holds:

Lemma 1 *For problem $F\mu||\sum|C_i - d|$ with a non-restrictive due date, the objective function value of the Kanet-schedule with the processing times on M_μ of the jobs as input data is a tight lower bound.*

Problem $O\mu||\sum|C_i - d|$ reduces, for a non-restrictive due date, to $Q\mu||\sum|C_i - d|$, which is polynomially solvable as well.[11]

[1] Slight modification of the notation in order to meet the usage in this article.

3 Basic Properties of Problems with Intermediate Storage Costs

In this section, we present some basic properties of multi-stage earliness-tardiness problems with intermediate storage costs. Furthermore, we present a simple enumerative algorithm.

3.1 Basic Properties

Lemma 2 *For problem* $F\mu||\sum|C_i-d|+\sum_{j=1}^{\mu-1}\sum_i(s_{i\,j+1}-c_{i\,j})$ *with non-restrictive due date, there is an optimal schedule with* $c_{\pi_m\,\mu}=d$, *where* $m=\lfloor\frac{n+1}{2}\rfloor$, *and* π *is the Kanet-order on machine* M_μ.

Remark: The proof is analogous to the one-machine problem. Note that $c_{\pi_m\,\mu}\in[d,d+p_{\pi_m\,\mu}]$ is sufficient if n is even. The sequences on $M_1,\ldots,M_{\mu-1}$ may be different from π. The lemma remains to be true for a job-dependent weighted penalty of the intermediate storage time.

Lemma 3 *For problem* $F\mu||\sum|C_i-d|+\sum_{j=1}^{\mu-1}\sum_i(s_{i\,j+1}-c_{i\,j})$ *with non-restrictive due date, there exists an optimal schedule where the early jobs on* M_μ *are processed without idle time.*

Remark: This is due to the fact that the additional penalty due to an increase of intermediate storage costs is exactly compensated by the decrease of the penalty caused by the deviation from the due date. Hence, the statemenent of Lemma 3 remains to be true for the problem $F\mu||\sum h_i|C_i-d|+\sum_{j=1}^{\mu-1}\sum_i w_{i\,j}(s_{i\,j+1}-c_{i\,j})$ with weighted linear penalties for the intermediate storage time and job-dependent penalties for the deviation from the due date as long as the weights $w_{i\,\mu-1}$ for $(s_{i\,\mu}-c_{i\,\mu-1})$ are lower than or equal to the weights h_i for $|C_i-d|$ for all jobs $i=1,\ldots,n$.

Property 1 *For problem* $F2||\sum|C_i-d|+\sum(s_{i\,2}-c_{i\,1})$ *and a non-restrictive due date* d, *there are instances for which no permutation schedule is optimal.*

Example 1 We consider the instance given in Table 1 with a non-restrictive due date.

jobs	1	2	3	4
M_1	6	15	10	2
M_2	6	6	9	1

Table 1: Processing times of Example 1

The best permutation schedules have the objective function value 25 and are given in Figures 1 and 2, but there exists (exactly) one non-permutation schedule with an objective function value of 24 as shown in Figure 3. Note that all three Gantt-charts illustrate only *one* of the various optimal schedules for the sequence π^2 on M_2, since for an even number n of jobs the optimal starting time of the middle job can vary within an interval (see remark after Lemma 2). Note further that the actual value of the due date does not influence the objective function value — it is sufficient that it is non-restrictive, which is surely the case for due dates greater than or equal to the sum of all processing times, which is in this example 55.

INSERT FIGURES 1-3 HERE

Property 2 *For problem* $F2||\sum|C_i - d| + \sum(s_{i\,2} - c_{i\,1})$ *with* $d = 0$*, there are instances where no permutation schedule is optimal.*

Example 2 Consider the job data given in Table 2. In all optimal schedules, the job sequence starts with (1,4,3,2) on M_1 and (1,2,3,4) on M_2.

M_i	1	2	3	4	$5, \ldots, 20$
M_1	1	3	4	5	100
M_2	29	31	33	36	100

Table 2: Processing times of Example 2

Lemma 4 *If there exists a schedule for problem* $F\mu||\sum|C_i - d| + \sum_{j=1}^{\mu-1}\sum_i(s_{i\,j+1} - c_{i\,j})$ *with a non-restrictive due date* d *without intermediate storage costs where the jobs on* M_μ *are ordered in Kanet-order without idle time, then this schedule is optimal.*

As a direct consequence, there exists the following polynomially solvable case: If d is non-restrictive and

$$\min_{i\in N}\{p_{i\,j+1}\} \geq \max_{i\in N}\{p_{i\,j}\},$$

for $j = 1, \ldots, \mu - 1$, then the Kanet-schedule on M_μ and $c_{i\,j} = s_{i\,j+1}$ $(j = 1, \ldots, \mu - 1)$ for all $i \in N$ is optimal.

3.2 Enumeration Algorithm for $F2||\sum|C_i - d| + \sum(s_{i\,2} - c_{i\,1})$

Based on the results given above, we are able to give a simple enumeration algorithm for problem $F2||\sum|C_i - d| + \sum(s_{i\,2} - c_{i\,1})$ with a non-restrictive due date d. Due to Property 1 it is necessary to distinguish the permutations π^1 and π^2 of the job set on M_1 and M_2, respectively. The following procedure is able to find a *best* schedule for given permutations π^1, π^2 and a non-restrictive due date. This *best* schedule is an optimal schedule for the problem if there is no other best schedule for different permutations π^1 and π^2 with a lower objective function value.

First, we define a start schedule as follows.

1. Initialization of the jobs on M_2: Let $c_{\pi^2_m\,2} := d$ (see Lemma 2). All (early and tardy) other jobs on M_2 are scheduled without idle time.

2. Initialization of the jobs on M_1: Let $c_{\pi^1_n\,1} := s_{\pi^1_n\,2}$. For $i = n-1$ to 1: Set $c_{\pi^1_i\,1} = \min\{s_{\pi^1_i\,2}, s_{\pi^1_{i+1}\,1}\}$.

If this start schedule has no intermediate storage costs, we can stop here, since this is already the best schedule. Otherwise, let i be the highest index of π^1 with $c_{\pi^1_i\,1} < s_{\pi^1_i\,2}$. Let j be the index of π^1 with

$$j = \min_{1\le k\le i}\left\{k | c_{\pi^1_l\,1} < s_{\pi^1_l\,2} \text{ for } k \le l \le i\right\}$$

and $\tilde{\imath} \le i$ be the index of π^2 with $\pi^2_{\tilde{\imath}} = \pi^1_i$. If $n + j - 1 \ge 2\tilde{\imath}$, we decrease i and repeat the test above. If $n + j - 1 < 2\tilde{\imath}$, we can decrease the objective function value by increasing the starting times of the jobs $\pi^1_j, , \ldots, \pi^1_n$ on M_1 and $\pi^2_{\tilde{\imath}+1}, \ldots, \pi^2_n$ on M_2 by

$$\delta = \min_{j\le k\le i}\left\{s_{\pi^1_k\,2} - c_{\pi^1_k\,1}\right\}.$$

We repeat this procedure *without* decreasing i. Due to the condition $n+j-1 < 2\tilde{\imath}$, we can stop the algorithm if $\tilde{\imath} = m - 1$. The inequality $n + j - 1 < 2\tilde{\imath}$ is derived by the following argumentation. Increasing the starting times of the jobs $\tilde{\imath}, \ldots, n$ on M_2 by one unit increases the objective function value by $2(n - \tilde{\imath})$ (deviation from the due date plus storage of intermediates). After this operation, increasing the starting times of the jobs $j, \ldots, n$ on M_1 by one unit reduces the costs by $n - j + 1$. Thus, we can reduce the overall objective function value through a combination of these two operations if $n - j + 1 > 2(n - \tilde{\imath})$.

Example 3 We illustrate the algorithm by the following example with seven jobs given in Table 3. We consider the permutations $\pi^1 = \pi^2 = (6, 3, 1, 4, 7, 5, 2)$. We get the schedule S^0 given in Figure 4.

jobs	1	2	3	4	5	6	7
M_1	9	13	16	16	12	20	6
M_2	17	11	16	2	12	17	2

Table 3: Processing times of Example 3

INSERT FIGURE 4 HERE

Schedule S^0 has intermediate storage costs. Hence, we try to reduce these costs. According to the algorithm described above, the highest index with $c_{\pi_i^1\,1} < s_{\pi_i^1\,2}$ is $i = 6$ $(= \tilde{\imath})$. Furthermore,

$$j = \min_{1 \le k \le 6} \left\{ k | c_{\pi_l^1\,1} < s_{\pi_l^1\,2} \text{ for } k \le l \le 6 \right\} = 1.$$

Since $n + j - 1 = 7 < 2\tilde{\imath} = 12$, we can reduce the objective function value through an increase of the starting times of job 2 on both machines and of the preceding jobs on M_1 by

$$\delta = \min_{1 \le k \le 6} \left\{ s_{\pi_k^1\,2} - c_{\pi_k^1\,1} \right\} = 1.$$

This leads to the schedule S^1 given in Figure 5.

INSERT FIGURE 5 HERE

Since now $c_{\pi_6^1\,1} = s_{\pi_6^1\,2}$, the algorithm reduces i by one. The algorithm determines for $i = 5 = \tilde{\imath}$ now $j = 1$, which leads to

$$\delta = \min_{1 \le k \le 5} \left\{ s_{\pi_k^1\,2} - c_{\pi_k^1\,1} \right\} = 5$$

and the schedule S^2 in Figure 6.

INSERT FIGURE 6 HERE

Still, there holds $c_{\pi_5^1\,1} < s_{\pi_5^1\,2}$, hence we calculate j without decreasing i. We get $j = 2$. Since $n + j - 1 = 8 < 2\tilde{\imath} = 10$, we get

$$\delta = \min_{2 \le k \le 5} \left\{ s_{\pi_k^1\,2} - c_{\pi_k^1\,1} \right\} = 1.$$

Again we have $c_{\pi_5^1\,1} < s_{\pi_5^1\,2}$. We get $j = 3$, $n + j - 1 = 9 < 2\tilde{\imath} = 10$, and $\delta = 4$, which yields schedule S^3 given in Figure 7. Now, we have $c_{\pi_5^1\,1} = s_{\pi_5^1\,2}$, and we

decrease i by one. We get $j = 3$. Since $n + j - 1 = 9 \geq 2\tilde{\imath} = 8$, we reduce i without the shift-operation, getting $i = 3 = m - 1$, and stop. Thus, Figure 7 displays the best schedule for the given permutations $\pi^1 = \pi^2 = (6, 3, 1, 4, 7, 5, 2)$.

INSERT FIGURE 7 HERE

This algorithm is able to solve problems with seven jobs in two minutes on an Intel Celeron with 466 MHz. The counterexample given in Table 1 has been found with the help of this algorithm.

4 The Flow Shop Problem with Intermediate Storage Costs

In this section, we will establish the NP-hardness in the strong sense of the problem $F2||\sum|C_i - d| + \sum(s_{i\,2} - c_{i\,1})$ with a non-regular objective function and a non-restrictive due date as well as for the corresponding problem $F2||\sum C_i + \sum(s_{i\,2} - c_{i\,1})$ with a regular objective function. The latter can also be seen as the problem $F2||\sum|C_i - d| + \sum(s_{i\,2} - c_{i\,1})$ with $d = 0$. Garey et al.[10] proved that $F2||\sum C_i$ is NP-hard in the strong sense. We will make use of their approach, but with several modifications and extensions. There are complexity results for infinite high intermediate storage costs, e.g. intermediate storage is forbidden. Gupta[12] showed that this problem is NP-hard in the strong sense by reducing the problem to a traveling salesman problem.

Note that it is not obvious that the problems considered in this article are NP-hard in the strong sense as well for two reasons. First, as we described in the preceding section, $F2||\sum|C_i - d|$ is polynomially solvable for non-restrictive d. Second, problem $F2||\sum(s_{i\,2} - c_{i\,1})$ is obviously polynomially solvable as well with the optimal objective function value of 0.

The recognition version of the 3-Partition problem which we will use in the complexity proofs can be formulated as follows: Given positive integers n, B, and a set of integers $A = \{a_1, a_2, \ldots, a_{3n}\}$ with $\sum_{i=1}^{3n} a_i = nB$ and $B/4 < a_i < B/2$ for $1 \leq i \leq 3n$, does there exist a partition $< A_1, A_2, \ldots, A_n >$ of A into 3-element sets such that $\sum_{a \in A_i} a = B$ for $i = 1, \ldots, n$?

Theorem 1 *The problem $F2||\sum|C_i - d| + \sum(s_{i\,2} - c_{i\,1})$ with a non-restrictive common due date is NP-hard in the strong sense.*

PROOF: The proof is done by reduction from 3-Partition. Suppose we are given n, B, and $A = \{a_1, a_2, \ldots, a_{3n}\}$ as specified in the 3-Partition problem. The corresponding input to the two-machine flow shop problem is derived in two main steps in the following way. First, we will find an instance of the problem for which we can calculate the optimal schedule and the optimal objective function

value easily; this will be called the skeleton schedule. This optimal schedule will define n time intervals around the due date where M_1 is idle. These *time slots* will all have the same appropriate length. In a second step, we add to the jobs of the skeleton schedule jobs which have to be processed in these time slots on M_1 in order to get a schedule with an objective function value lower than a certain bound. We will show, too, that this filling of the time slot is possible if and only if 3-Partition has a solution.

We introduce the following quantities:

$$
\begin{aligned}
m &= \left\lfloor \frac{n+1}{2} \right\rfloor \\
k &> nB \\
v &> 4n(k+3)B \qquad (4) \\
r &= v(k+3) + B \\
G_1 &= \frac{k(k-1)}{2} v + 3kv \qquad (5) \\
G_2 &= (k+3)r(m(m-1) + m(n+1-2m)) \qquad (6) \\
G &= nG_1 + nB + G_2 \\
x &> 2G \\
o &> 2G, \ o \text{ even} \qquad (7) \\
f &> 2G
\end{aligned}
$$

Each job i is characterized by the ordered pair of its processing times $(p_{i\,1}, p_{i\,2})$ on M_1 and M_2, respectively. The input data for the skeleton schedule, which is a part of the whole instance we have to construct to prove the complexity, consists of the following $o + n + f + 2$ jobs:

1. job set $\mathcal{O} = \{O_i | O_i = (x, x) \quad \text{for} \quad i = 1, \ldots, o\}$,
2. job set $\mathcal{L} = \{L_i | L_i = (0, r) \quad \text{for} \quad i = 1, \ldots, n\}$,
3. job $U = (x, 0)$,
4. job $W = (0, x)$,
5. job set $\mathcal{F} = \{F_i | F_i = (0, 0) \quad \text{for} \quad i = 1, \ldots, f\}$.

First, we describe the skeleton schedule, and then we explain why this schedule is optimal, and in which sense it is unique. The optimal schedule for the instance given above is a permutation schedule which is not true in general (see Property 1). We set for the jobs in $\mathcal{L}$ $c_{L_g\,2} = d + r(g - m)$ and $c_{L_g\,1} = s_{L_g\,2}$ for all $g = 1, \ldots, n$. Each job g of $\mathcal{L}$ constitutes a slot, e.g. a time interval I_g:

$$
I_g = \begin{cases} [s_{L_g\,1}, \min\{c_{L_g\,2}, s_{L_g+1\,1}\}], & g \in \{1, 2, \ldots, n-1\} \\ [s_{L_g\,1}, \min\{c_{L_g\,2}, s_{W\,1}\}], & g = n \end{cases} \qquad (8)
$$

The n jobs scheduled in the way given above yield our desired n time slots, each of length $v(k+3)+B$ on M_1.

To keep the formulae short, let $\sigma = d - rm$. Immediately before the first job of $\mathfrak{L}$ is processed, job U is scheduled with $c_{U\,1} = s_{U\,2} = c_{U\,2} = \sigma$. Immediately after the last job of $\mathfrak{L}$, job W is scheduled with $s_{W\,1} = c_{W\,1} = s_{W\,2} = \sigma + rn$. The jobs of $\mathcal{O}$ are scheduled symmetrically around the jobs scheduled so far ($o = |O|$ is an even number, see Inequality (7)), that means:

1. $c_{O_{o/2}\,2} = s_{U\,2}$ and $c_{O_{o/2}\,1} = s_{O_{o/2}\,2}$; $c_{O_i\,2} = s_{O_{i+1}\,2}$ and $c_{O_i\,1} = s_{O_{i+1}\,1}$ for $i = o/2-1, o/2-2, \ldots, 1$.

2. $s_{O_{o/2+1}\,2} = c_{V\,2}$ and $s_{O_{o/2+1}\,1} = s_{V\,2}$; $s_{O_{i+1}\,2} = c_{O_i\,2}$ and $s_{O_{i+1}\,1} = c_{O_i\,1}$ for $i = o/2+1, o/2+2, \ldots, o-1$.

The jobs of $\mathcal{F}$ start and end at the due date d. Thus, there is no idle time on M_2 between the processing of the jobs, even for the tardy ones. A sketch of the skeleton schedule is given in Figure 8. Each box in the Gantt-charts given in the remainder of the paper contains the processing time of the job and, where necessary, the corresponding job in parantheses.

INSERT FIGURES 8-9 HERE

The optimality of this schedule can be seen by the following argumentation. First, the starting time on M_2 is the completion time on M_1 for every job in this schedule. Thus, there are no intermediate storage costs at all. Furthermore, the jobs on M_2 are ordered "nearly" according to the Kanet-order which is optimal for the corresponding one-machine problem. The only exception is the scheduling of U. Since the processing time of this job on M_2 is zero, it would be scheduled at time d in the optimal one-machine schedule. So the whole difference between the value of the schedule described above and the lower bound according to Lemma 1 is given by $d - c_{U\,2}$, which is mr. Without an alteration of $c_{U\,1}$, the schedule with $c_{U\,2} = d$ has the same objective function value as the one described above, since the decrease in the penalty due to the deviation from the due date is exactly compensated by the increase of the penalty $c_{U\,1} - s_{U\,2}$. Hence, there are $m+1$ (as will be clear soon optimal) schedules only differing in $s_{U\,2}$: $s_{U\,2} = \sigma + ir$ for $i = 0, \ldots, m$. The *only* possibility remaining which could be able to reduce the objective function value further is a change in the starting time of $s_{U\,2}$ within the interval $(\sigma, \sigma + nr)$ together with an appropriate increase of $s_{U\,1}$ towards the due date. Clearly, the maximum additional decrease of the objective function value is mr. In order to increase $s_{U\,1}$, we have to change the starting times of some jobs of set $\mathfrak{L}$ on M_1 as well. Since o is greater than $2G$, increasing the starting time of any job of $\mathfrak{L}$ even by only one unit would increase the penalty for the jobs of $\mathcal{O}$ by more than $G > mr$. But the starting times on M_1 of the jobs of $\mathfrak{L}$ cannot be decreased either, since this can only be achieved by interchanges of some jobs

in $\mathcal{L}$ with U on M_1. This would increase the objective function value by at least $x - 2r > G$ for the case of exchanging the order of processing on M_1 between U and L_1.

Without the job set $\mathcal{F}$, a shift of all starting times of all jobs on both machines by 1 increases the objective function value by only 1 if n is odd and not at all if n is even. This is due to the fact that all increased tardiness penalties are compensated by the decreased earliness penalties except for the job L_m for odd n, which was punctual before and would now be penalized by $d + 1 - d = 1$. Due to the jobs in $\mathcal{F}$, the objective function value is increased by f additionally by performing a shift of the schedule by one time unit. If n is even, a shift of all starting times except for the jobs of $\mathcal{F}$ on both machines by r is possible, but this does neither change the objective function value nor the argumentation in the remainder of the proof.

The schedule constructed so far is unique apart from permutations of the jobs within their job set $\mathcal{O}$ or $\mathcal{L}$ and the described modifications of the starting time $s_{U\,2}$ (and the shift by r if n is even). These in total $o!l!(m+1)$ for odd and $2o!l!(m+1)$ for even n optimal schedules are regarded as equivalent. The member of this set of equivalent schedules which was defined above is referred to as the skeleton schedule in the remainder.

Let the objective function value of this optimal schedule be Y. Now add the following $(3+k)n$ jobs to the present job set:

1. $\mathcal{Q} = \{Q_i | Q_i = (v + a_i, 0) \quad \text{for} \quad i = 1, \ldots, 3n\}$
2. $\mathcal{V} = \{V_i | V_i = (v, 0) \quad \text{for} \quad i = 1, \ldots, kn\}$

If a solution of 3-Partition exists, the n slots on M_1 of the skeleton schedule of length $v(k+3)+B$ can be filled exactly. We number these slots according to the constituting L-job from 1 to n. Consider one such slot $g \in \{1, \ldots, n\}$. The indices of the three $\mathcal{Q}$-jobs contained are $\lambda_g(1)$, $\lambda_g(2)$, $\lambda_g(3)$ with $a_{\lambda_g(1)} \leq a_{\lambda_g(2)} \leq a_{\lambda_g(3)}$ and $\sum_{i=1}^{3} a_{\lambda_g(i)} = B$ for all $g \in \{1, \ldots, n\}$. The jobs of each filled slot g are executed on M_2 at the completion time $c_{L_g\,2}$ of the constituting $\mathcal{L}$-job g. The jobs in the slot are assigned according to the LPT-rule (see Figure 9). The resulting contribution IS to the objective function value due to the intermediate storage costs contains the values of $a_{\lambda_g(1)} \leq a_{\lambda_g(2)} \leq a_{\lambda_g(3)}$ explicitly. An upper bound without these values can be derived as follows:

$$\begin{aligned} IS &= \frac{k(k-1)}{2}v + kv + (kv + a_{\lambda_g(1)}) + (kv + a_{\lambda_g(1)} + a_{\lambda_g(2)}) \\ &\leq \frac{k(k-1)}{2}v + 3kv + B = G_1 + B, \end{aligned}$$

with G_1 as given in Equation (5). Additionally, we have the penalty due to the deviation of the completion time of the jobs of $\mathcal{Q}$ and $\mathcal{V}$ on M_2 from the due date. For each slot g, this completion time is $\sigma + gr$ for all the contained $k+3$ jobs,

thus G_2 (see Equation (6)) is the total contribution to the objective function value caused by the deviation of the completion time of the jobs in the slots from the due date. Summing up the penalties of the skeleton schedule, the upper bound G_1 for the intermediate storage costs of each of the n slots, and G_2, we get the following upper bound for the optimal objective function value of the complete instance:

$$Z = Y + nG_1 + G_2 + nB. \tag{9}$$

Hence, if 3-Partition has a solution, there exists a schedule with an objective function value no greater than Z. In order to complete the proof, we have to assure that the existence of such a schedule is sufficient for the existence of a solution of 3-Partition.

It is clear that the filled time slots can have the same length $v(k+3)+B$ only if 3-Partition has a solution. Thus, it remains to show that in any schedule with an objective function value lower than or equal to Z, all n time slots must have the length $v(k+3)+B$.

From the discussion above it is clear that the contribution Y of the skeleton schedule is the minimal contribution to the overall objective function value. Also, we can exclude modifications, based on a shift of the starting times of all jobs by the same amount, since this does not change the size of the slots. Modifications of the order of processing of the jobs of the skeleton schedule which do not lead to one of the equivalent schedules described above will increase the contributions of the jobs of the skeleton schedule by more than G and thus lead to an objective function value significantly higher than Z.

Obviously, no job of $\mathcal{V}$ or $\mathcal{Q}$ can be completed after $c_{L_n\,2}$, since the additional penalty due to the tardiness for this job is at least x. As well, no job of $\mathcal{V}$ or $\mathcal{Q}$ can be completed before $c_{L_1\,2}$.

Furthermore, no job of $\mathcal{V}$ or $\mathcal{Q}$ can be started before $s_{L_1\,1}$ on M_1, since there are only two possibilities of starting a job of $\mathcal{V}$ or $\mathcal{Q}$ before $s_{L_1\,1}$. First, we can exclude an exchange with any job of the skeleton schedule which is processed before $s_{L_1\,1}$, since this would increase the costs of intermediate storage by more than G. The second possibility is a decrease of the starting times of all jobs of the skeleton schedule on M_1 which are processed before $s_{L_1\,1}$ by a value δ. But this would increase the according intermediate storage costs of these skeleton jobs by $(o/2+1)\delta > \delta G$ ($o/2$ jobs of $\mathcal{O}$ and the job U). Thus, the processing of all additional jobs has to occur within the time window $[\sigma, \sigma + nr]$ in order to get a schedule with an objective function value no greater than Z. Note that the jobs of $\mathcal{V}$ and $\mathcal{Q}$ which are started on M_1 before d do not need to be completed on M_2 exactly when the constituting job of $\mathcal{L}$ is completed on M_2, but can be shifted to any other constituting job of a following slot before or at the due date since the additional costs of the intermediate storage are exactly compensated by the reduction of the earliness. Nevertheless, at least the additional jobs in the slot

which is completed on time (precisely at the due date) have to be completed on M_2 exactly at d as well.

It remains to be shown that all slots I_g ($g = 1, \ldots, n$, see Equation (8)) must have the same length $v(k+3)+B$ in order to get an objective function value no greater than Z. All additional jobs are to be scheduled within the time window $[\sigma, \sigma + nr]$. We will now prove that every schedule with this property and one slot containing more than $k+3$ jobs can be transformed in polynomial time into a schedule with an objective function value at least $2n(k+3)B$ lower in which every slot contains exactly $k+3$ jobs.

The algorithm is as follows. We start with the slot I_g with the highest index g which contains more than $k+3$ jobs. If the number of jobs is higher than $k+4$, we exchange the order of processing of job L_g and the job of set $\mathcal{V}$ or $\mathcal{Q}$ on M_1, which is processed immediately after job L_g, until the number of jobs in the slot is $k+4$. Clearly, every exchange reduces the objective function value by at least v. If the number of jobs is $k+4$, there can occur two cases. Let b be the job following immediately after job L_g on M_1. If $c_{b\,1} \leq s_{L_g\,2}$, we can exchange the order of processing of jobs L_g and b on M_1. By doing so, we decrease the objective function value by at least v. In the second case, $c_{b\,1} > s_{L_g\,2}$. Hence, there are jobs of $\mathcal{Q}$ which are processed before L_g on M_1. Our aim is to decrease $c_{b\,1}$ such that $c_{b\,1} \leq s_{L_g\,2}$. This is done by pairwise interchanges of $\mathcal{V}$-jobs (different from b) in slot g with $\mathcal{Q}$-jobs in slots which are processed before slot g. Since the processing time of a job of $\mathcal{Q}$ is bigger than v, the costs of intermediate storage increase for *all* jobs in-between interchanged jobs. It follows that in the worst case *all* jobs of $\mathcal{Q}$ which are processed before L_g are in slot 1 and $g = n$. Since $a_i > B/4$, we will at the most need four such interchanges. An upper bound for the total increase of the objective function value due to intermediate storage is $2n(k+3)B$ ($a_i < B/2$). The scanning and the pairwise interchanges for all the at the most four jobs of $\mathcal{Q}$ can be done in $O(n)$ time. After these operations, we have $c_{b\,1} \leq s_{L_g\,2}$. Hence, we can reduce the objective function value by at least v by interchanging job L_g with job b, which leads to an overall reduction of the objective function value compared with the original schedule of at least $2n(k+3)B$ due to Inequality (4).

Thus, without loss of generality, we only need to consider schedules with exactly $k+3$ jobs in the slots. Note that in Equation (9) the contribution nG_1+G_2 is the penalty for the jobs processed in the slots if all $k+3$ jobs have the *same* processing time v on M_1. This means that $Y + nG_1 + G_2$ is a lower bound for the optimal objective function value because the processing time of the jobs of $\mathcal{Q}$ is greater than v.

In order to complete the proof, we will show that a slot with a length greater than $v(k+3)+B$ leads to an objective function value which is more than nB greater than the one we defined under the assumption that the instance of 3-partition has a solution. If one slot contains a total processing time on M_1 of $\delta + r$, then the starting time of the constituting job of $\mathcal{L}$ on M_1 is decreased by δ.

This results in additional intermediate storage costs for the jobs of the preceding slot of $\delta(k+3) > nB$. ∎

Corollary 1 *Problem* $F2||\sum C_i + \sum(s_{i\,2} - c_{i\,1})$ *is NP-hard in the strong sense.*

This follows as a direct consequence from the proof of Theorem 1, since the tardy jobs in the optimal schedule given above are obviously a suitable instance to establish the NP-hardness in the strong sense of problem $F2||\sum C_i + \sum(s_{i\,2} - c_{i\,1})$ with a regular optimization problem.

5 The Open Shop Problem with Intermediate Storage Costs

In this section, we will prove that the analogous open shop problem is NP-hard in the strong sense for $d = 0$ and a non-restrictive due date. The notation for the intermediate storage costs has to be slightly changed for the open shop problem. With $s_{i\,[2]} - c_{i\,[1]}$, we denote the time between the completion of the first *operation* and the start of the second *operation*, since the order of processing can be chosen arbitrarily.

5.1 NP-Hardness for $d = 0$

Theorem 2 *The problem* $O2||\sum |C_i - d| + \sum(s_{i\,[2]} - c_{i\,[1]})$ *with a common due date* $d = 0$ *is NP-hard in the strong sense.*

Proof: Again, we prove the claim by reduction from 3-Partition in the formulation given in Section 4. First, we define the following values:

$$
\begin{aligned}
v &> 4nB \\
t &= 3v + B \\
P\tilde{V} &= 4nv + \left(\frac{3}{2}n(n-1) + n\right) t \qquad (10) \\
PT &= \frac{n(n+1)}{2} t \qquad (11) \\
x &> PT + P\tilde{V} \\
PX &= 2ntx + x^2(x+1) \qquad (12)
\end{aligned}
$$

Given an instance of 3-Partition, we construct an instance of the scheduling problem with $4n + 2x$ jobs divided into the following sets:

1. $\mathcal{T} = \{T_i | T_i = (t, 0) \quad \text{for} \quad i = 1, \ldots, n\}$,
2. $\mathcal{X}1 = \{X1_i | X1_i = (x, 0) \quad \text{for} \quad i = 1, \ldots, x\}$,

3. $\mathcal{X}2 = \{X2_i | X2_i = (0, x) \quad \text{for} \quad i = 1, \ldots, x\}$,

4. $\mathcal{V} = \{V_i | V_i = (0, a_i + v) \quad \text{for} \quad i = 1, \ldots, 3n\}$.

The recognition problem is: Given the above instance for $O2||\sum |C_i - d| + \sum(s_{i\,[2]} - c_{i\,[1]})$, does there exist a schedule with an objective function value lower than $P\tilde{V} + 2nB + PX + PT$? We will now define such a schedule under the assumption that 3-Partition has a solution. Without loss of generality, this solution is of the following form for all $i = 1, \ldots, n$:

$$a_i + a_{i+n} + a_{i+2n} = B \tag{13}$$

$$a_i \leq a_{i+n} \leq a_{i+2n} \tag{14}$$

The partial sequence $(i, i+n, i+2n)$ is called i-th triple. Obviously, the jobs of sets $\mathcal{X}1$ and $\mathcal{X}2$ have to be scheduled last in order to get a low objective function value. The other jobs must be processed without idle time in the interval $[0, nt]$. The schedule is as follows ($i = 1, \ldots, n$ and $j = 1, \ldots, x$):

$$\begin{aligned}
s_{V_i\,1} &= c_{V_i\,1} = s_{V_i\,2} = (i-1)t \\
s_{V_{i+n}\,1} &= c_{V_{i+n}\,1} = (i-1)t \\
s_{V_{i+n}\,2} &= c_{V_i\,2} \\
s_{V_{i+2n}\,1} &= c_{V_{i+2n}\,1} = it \\
c_{V_{i+2n}\,2} &= it
\end{aligned}$$

$$\begin{aligned}
s_{T_i\,1} &= (i-1)t \\
s_{T_i\,2} &= it \\
s_{X2_j\,2} &= s_{X1_j\,1} = nt + (j-1)x \\
s_{X1_j\,2} &= s_{X2_j\,1} = nt + jx
\end{aligned}$$

A sketch can be found in Figure 10.

INSERT FIGURE 10 HERE

Obviously, a schedule is not uniquely determined by the above scheduling procedure. For instance, we can change the sequence $(V_1, V_{n+1}, V_{2n+1}, V_2, V_{n+2}, V_{2n+2})$ into $(V_2, V_{n+2}, V_{2n+2}, V_1, V_{n+1}, V_{2n+1})$. Thus, as in the NP-hardness proof for $F2||\sum |C_i - d| + \sum(s_{i\,2} - c_{i\,1})$, the schedule defined above is a representative of the set of all possible schedules of this form. This is of no importance for the proof, just as little as the fact that the objective function value is lower if the job with longest processing time of a triple i, job V_{i+2n}, is processed in-between jobs V_i and V_{i+n}.

The contributions of $\mathcal{X}1, \mathcal{X}2$, and $\mathcal{T}$ to the objective function value of this schedule are PX and PT as given in Equations (12) and (11). The penalties Q_1, Q_2, Q_3 for the jobs $V_1, \ldots, V_n$, $V_{n+1}, \ldots, V_{2n}$, $V_{2n+1}, \ldots, V_{3n}$, of $\mathcal{V}$ are

$$Q_1 = \frac{n(n-1)}{2}t + nv + \sum_{i=1}^{n} a_i \tag{15}$$
$$< \frac{n(n-1)}{2}t + nv + \frac{nB}{2}$$
$$Q_2 = \frac{n(n-1)}{2}t + 3nv + \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} a_i + \sum_{i=n+1}^{2n} a_i \tag{16}$$
$$< \frac{n(n-1)}{2}t + 3nv + \frac{3nB}{2}$$
$$Q_3 = \frac{n(n+1)}{2}t \tag{17}$$

We will make use of the following lower bound $P\tilde{V}$ (see Equation (10)) for the contribution of the jobs of set $\mathcal{V}$ to the objective function value. This bound is the value for scheduling $3n$ *identical* jobs of a set $\tilde{\mathcal{V}}$ consisting of $\tilde{V}_i = (0, v)$ for $i = 1, \ldots, 3n$ in the gaps made by the jobs $T_i = (t, 0)$ for $i = 1, \ldots, n$: Without loss of generality, we assume that the jobs are numbered according to the order of completion. Let us start with the last job processed of $\tilde{\mathcal{V}}$. This job $\tilde{V}_{3n}$ can be completed on M_2 in the interval $[nt - nB, nt]$. Obviously, the completion time nt for $\tilde{V}_{3n}$ on both machines yields the lowest contribution to the objective function value of this job. The second and third last jobs $\tilde{V}_{3n-1}$ and $\tilde{V}_{3n-2}$ can be completed in the intervals $[nt - nB - v, nt - v]$ and $[nt - nB - 2v, nt - 2v]$, respectively. Obviously, the lowest contribution to the objective function value of $\tilde{V}_{3n-1}$ and $\tilde{V}_{3n-2}$ is achieved if $s_{\tilde{V}_{3n-2}\,1} = s_{\tilde{V}_{3n-2}\,2} = (n-1)t$, $s_{\tilde{V}_{3n-1}\,1} = (n-1)t$ and $s_{\tilde{V}_{3n-1}\,2} = (n-1)t + v$. The same argumentation can be applied to the jobs $\tilde{V}_{3(n-1)}, \tilde{V}_{3(n-1)-1}$ and $\tilde{V}_{3(n-1)-2}$ and so on. The difference between $P\tilde{V}$ and $Q_1 + Q_2 + Q_3$ is less than $2nB$.

It remains to be shown that there is no schedule with an objective function value less than $P\tilde{V} + 2nB + PX + PT$ if 3-Partition has no solution. In this case, there is a $k \in [1, \ldots, n]$ to which there is one job l of $\mathcal{V}$ with $s_{l\,2} < tk$ and $c_{l\,2} > tk$. Since $nt - 3nv = nB$, there can occur only two cases.

The first possibility is $0 < \delta := tk - s_{l\,2} < nB$. Since a job cannot be processed simultaneously on both machines, job l is processed at time $t(k-1)$ on M_1 to get a low objective function value, which leads to intermediate storage costs of $t - \delta$. Job l is completed at time $tk + v + a_l - \delta$. Summing up, we get as a lower bound of the contribution of job l to the objective function value:

$$t(k+1) + v + a_l - 2\delta > t(k+1) + v - 2nB.$$

The total penalty for the other jobs in $\mathcal{V}$ is necessarily no less than $P\tilde{V}-tk-v$. Hence, we get as a lower bound for the overall objective function value of such a schedule

$$PX + PT + P\tilde{V} + t - 2nB > PX + PT + P\tilde{V} + 2nB,$$

and thus we can conclude that in the case of $0 < \delta := tk - s_{l\,2} < nB$, there is no schedule leading to an objective function value lower than $PX+PT+P\tilde{V}+2nB$.

The second possibility is $0 < \delta := c_{l\,2} - tk < nB$. The lowest penalty for this job is obtained by processing job l at time $t(k-1)$ on M_1, causing intermediate storage costs of $t - v - a_l + \delta$. Job l is completed at time $tk + \delta$, hence the processing of this job is penalized in total by $t(k+1) - v - a_l + 2\delta$. A lower bound for the total penalty of the remaining jobs of $\mathcal{V}$ is $P\tilde{V} - tk$. It follows that a lower bound for such a schedule is

$$PX + PT + P\tilde{V} + t - v - a_l + 2\delta > PX + PT + P\tilde{V} + 2nB. \qquad (18)$$

Thus, there exists a schedule with an objective function value less than $PX + PT + P\tilde{V} + 2nB$ if and only if 3-Partition has a solution. ∎

5.2 NP-Hardness for Non-Restrictive *d*

Finally, we will prove that the same result holds for a non-restrictive due date. This proof is more difficult, so we will give a sketch of the argumentation first.

As in the proof for the flow-shop problem, we will introduce a job set $\mathcal{F}$ to guarantee that some of the jobs have to be completed exactly at the due date. However, it is impossible to create time slots as in the proof for the flow-shop problem: If one increases the starting time of the second operation of an early job, the additional penalty due to an increase of intermediate storage costs is exactly compensated by the decrease of the penalty caused by the deviation from the due date. Thus, all operations of the early jobs with zero processing time can be scheduled at time d without increasing the objective function value. Time slots can be created only for *tardy* jobs. It suggests itself that we use the same job set $\mathcal{V}$ derived from an instance of 3-Partition as in the proof for problem $O2|d_i = 0|\sum |C_i - d| + \sum(s_{i\,[2]} - c_{i\,[1]})$. This job set will be accompanied by a job set $\mathcal{W}$ the elements of which cause similar costs for the deviation from the due date but higher costs due to intermediate storage. This forces that an early processing of the $\mathcal{W}$-jobs is preferred to an early processing of the $\mathcal{V}$-jobs which are derived from an instance of 3-Partition. Finally, we will need more than $3n$ jobs of job set $\mathcal{W}$. This is due to the fact that the penalty for a job of set $\mathcal{W}$ processed early is less than for a tardy job with the same deviation from the due date.

Theorem 3 *The problem* $O2||\sum |C_i - d| + \sum(s_{i\,[2]} - c_{i\,[1]})$ *with a non-restrictive due date is NP-hard in the strong sense.*

PROOF: The proof is composed of the following parts. The Basic Definitions define the instance of 3-Partition and several values like Z and Δ. The Basic Properties will define a set of *normal* schedules, which contains all candidates for a yes-answer of the recognition problem of the form: "Does there exist a schedule with an objective function value lower than $Z + \Delta$?" Part 1 will show that only schedules with at least n tardy $\mathcal{T}$-jobs have to be considered. In Part 2, we will prove that the objective function value of all schedules with $3k$ tardy jobs of $\mathcal{V}$ is more than Δ lower than a certain value if and only if the 3-Partition subproblem with k triples has a solution.

We will show in Part 3 that the objective function value of a schedule based on a solution of the 3-Partition problem with $k + 2$ ($\leq n$) triples is more than Δ lower than any schedule basing on the solution of the 3-Partition problem with only k triples. Part 4 shows that every solution as defined before on the basis of a solution of 3-Partition has an objective function value lower than $Z + \Delta$.

5.2.1 Basic Definitions: The Recognition Problem

Again, we prove the claim by reduction from 3-Partition in the formulation given in Section 4. Without loss of generality, let n be even. (If n is odd, we add the three jobs $(0, v)$, $(0, v)$, and $(0, v + B)$ to the job set $\mathcal{V}$.) We define the following positive integer values:

$$\gamma \geq 3 \tag{19}$$

$$\Delta = 20n^2B \tag{20}$$

$$\alpha > \gamma nB \tag{21}$$

$$v > 4\Delta\alpha^2$$

$$t = 3v + B$$

$$PEW(q) = \frac{q^2}{4}(18v - 15\alpha + B) - \frac{q}{2}(3v + B) \tag{22}$$

$$PET(q) = \frac{q^2}{2}t - \frac{q}{2}t \tag{23}$$

$$PTT(q) = \frac{q^2}{2}t - \frac{q}{2}(2\gamma n + 1)t + \frac{\gamma^2 n^2 + \gamma n}{2}t \tag{24}$$

$$\begin{aligned} PTW(q) = {} & \frac{3}{2}q^2t + \frac{q}{2}\left(6t - 25v - 6(\gamma - 1)nt - 5\alpha - 10B\right) + \frac{25}{2}vn(\gamma - 1) \\ & + 5(\gamma - 1)nB + \frac{5}{2}(\gamma - 1)n\alpha - 3(\gamma - 1)nt + \frac{3}{2}(\gamma - 1)^2n^2t \end{aligned} \tag{25}$$

$$PTV(q) = -3ntq + 3(\gamma - 1)n^2t + \frac{3}{2}n(n - 1)t + 7nv + 4nB + n^2B + nt \tag{26}$$

$$x > 8\gamma^2n^2t$$

$$PX = 2\gamma ntx + 4x^3 \tag{27}$$

The jobs are given by the following sets:

1. $\mathcal{T} = \{T_i | T_i = (t, 0) \quad \text{for} \quad i = 1, \ldots, \gamma n\}$,
2. $\mathcal{X} = \{X_i | X_i = (x, x) \quad \text{for} \quad i = 1, \ldots, 4x\}$,
3. $\mathcal{V} = \{V_i | V_i = (0, a_i + v) \quad \text{for} \quad i = 1, \ldots, 3n\}$,
4. $\mathcal{W}1 = \{W1_i | W1_i = (0, v - \alpha) \quad \text{for} \quad i = 1, \ldots, \frac{5}{2}(\gamma - 1)n\}$.
5. $\mathcal{W}2 = \{W2_i | W2_i = (0, v + 5\alpha + 2B) \quad \text{for} \quad i = 1, \ldots, \frac{1}{2}(\gamma - 1)n\}$
6. $\mathcal{W} = \mathcal{W}1 \cup \mathcal{W}2$
7. $\mathcal{F} = \{F_i | F_i = (0, 0) \quad \text{for} \quad i = 1, \ldots, PX\}$.

Jobs are called early if they are completed before or at the due date, and tardy otherwise. We will use the indices ϵ and τ in the remainder to refer to the subset of jobs scheduled early or tardy. For instance, $\mathcal{T}_\epsilon$ is the subset of jobs of set $\mathcal{T}$ which are scheduled early in a certain schedule. We will show that there exists a schedule with an objective function value less than $Z + \Delta$ where

$$Z = \min_{q=1,\ldots,(\gamma-1)n/2} \{\min\{LB(0, q), LB(1, q)\}\} + PX$$

if and only if 3-Partition has a solution where $LB(k, q)$ is defined by

$$\begin{aligned} LB(k,q) = & \underbrace{\sum_{i=0}^{\frac{5(q-n+k)}{2}-1} (v-\alpha)i}_{\text{early jobs of } \mathcal{W}1} \\ & + \underbrace{\sum_{i=0}^{3(n-k)-1} iv + \frac{15}{2}(n-k)(q-n+k)(v-\alpha)}_{\text{early jobs of } \mathcal{V}} \\ & + \underbrace{\sum_{i=0}^{\frac{q-n+k}{2}-1} (v+5\alpha+2B)\, i + \left(\frac{5}{2}(q-n+k)(v-\alpha) + (n-k)t\right) \frac{q-n+k}{2}}_{\text{early jobs of } \mathcal{W}2} \\ & + \underbrace{\sum_{i=0}^{q-1} ti}_{\text{early jobs of } \mathcal{T}} \qquad + \underbrace{\sum_{i=1}^{\gamma n-q} ti}_{\text{tardy jobs of } \mathcal{T}} \\ & + \underbrace{\frac{1}{2}(25v+5\alpha+10B)(\gamma n-q-k) + 12t \sum_{i=0}^{\frac{\gamma n-q-k}{2}-1} i}_{\text{tardy jobs of } \mathcal{W}} \end{aligned} \tag{28}$$

$$\underbrace{+ 3kt(\gamma n - q - k) + 4kv + kt + \sum_{i=0}^{k-1} 3it}_{\text{tardy jobs of } \mathcal{V}}$$

5.2.2 Basic Properties: The Normal Schedule

As in the preceding proofs, the jobs of $\mathcal{F}$ guarantee that one job of $\mathcal{T}$ is completed *on time* (precisely at the due date) in a schedule with an objective function value lower than $Z + \Delta$. As well, PX is the minimal contribution of the jobs of $\mathcal{X}$ under the condition that the other jobs are in a time interval γnt around the due date. We have for the contribution of the tardy jobs of $\mathcal{X}$:

$$2(\gamma n - |\mathcal{T}_\epsilon|)tx + 2x \cdot 2 \cdot \sum_{i=1}^{x} i = 2(\gamma n - |\mathcal{T}_\epsilon|)tx + 2x^2(x+1).$$

For the early jobs of $\mathcal{X}$ we get

$$2|\mathcal{T}_\epsilon|ntx + 2x \cdot 2 \cdot \sum_{i=1}^{x-1} i = 2|\mathcal{T}_\epsilon|tx + 2x^2(x-1).$$

This adds up to PX as given in Equation (12). It follows that there is no idle time in-between the completion of the last early job of $\mathcal{X}$ on M_2 and the due date. The same is true for the time in-between d and the starting time of the first tardy job on M_2 of $\mathcal{X}$. We get the following equation:

$$(3|\mathcal{T}_\epsilon| - |\mathcal{W}2_\epsilon| - |\mathcal{W}1_\epsilon| - |\mathcal{V}_\epsilon|)v = \sum_{i \in \mathcal{V}_\epsilon} a_i - \alpha|\mathcal{W}1_\epsilon| + (5\alpha + 2B)|\mathcal{W}2_\epsilon| - |\mathcal{T}_\epsilon|B.$$

The right side is less than v. Thus, we get the two equations

$$\sum_{i \in \mathcal{V}_\epsilon} a_i - \alpha|\mathcal{W}1_\epsilon| + (5\alpha + 2B)|\mathcal{W}2_\epsilon| = |\mathcal{T}_\epsilon|B \tag{29}$$

$$|\mathcal{W}2_\epsilon| + |\mathcal{W}1_\epsilon| + |\mathcal{V}_\epsilon| = 3|\mathcal{T}_\epsilon| \tag{30}$$

Since $|\mathcal{T}_\epsilon|B \leq \gamma nB < \alpha$, we get from Equations (29) and (30):

- the number of early jobs of $\mathcal{W}1$ is a multiple of 5;
- the number of early jobs of $\mathcal{W}2$ is exactly one fifth of the number of early jobs of $\mathcal{W}1$;
- the number of early jobs of set $\mathcal{W}$ is a multiple of six;

- the number of early jobs of $\mathcal{V}$ is a multiple of 3, and the a_i of the jobs processed early sum up to a multiple of B.

The analogous statements for the tardy jobs of sets $\mathcal{V}$ and $\mathcal{W}$ hold, too. We assume in the following that the tardy jobs of sets $\mathcal{V}$ and $\mathcal{W}$ are processed on M_1 such that the intermediate storage costs are minimal for a given sequence of the jobs of $\mathcal{V}$ and $\mathcal{W}$. It is obvious that the contribution of the early jobs of $\mathcal{V}$ and $\mathcal{W}$ is minimal if they are scheduled in LPT-order on M_2 and at time d on M_1. We call a schedule with the properties described so far a *normal* schedule. We can conclude already that the terms given in Equation (28) describing the contribution of the early jobs are indeed a lower bound, where q is the number of early jobs of $\mathcal{T}$ and $3k$ is the number of tardy jobs of $\mathcal{V}$. Note further that, due to the conclusions from Equations (29) and (30), the upper summation indices given in Equation (28) are integer in a normal schedule, since $(q-(n-k)) = 2|\mathcal{W}2|$.

5.2.3 Part 1: $|\mathcal{T}_\tau| \geq n$

The optimal number of jobs processed before the due date in general depends on the chosen sequence. First, we will show that we can decrease the objective function value of every normal schedule with less than n tardy jobs of $\mathcal{T}$ by increasing the starting times of all jobs (except those of $\mathcal{F}$) by a value such that n or more jobs are tardy. This is important in order to guarantee that all jobs of $\mathcal{V}$ can be scheduled tardy.

Denote with q the number of early jobs of $\mathcal{T}$. Given the sequences for M_1 and M_2 for all jobs defined in the problem formulation, we get $\gamma n+1$ different normal schedules which can be classified by $q = 0, \ldots, \gamma n$. Especially, the objective function value of a normal schedule can be regarded as a function of q. We will develop an upper bound for q^*, the value which minimizes the objective function value for the given sequences. To achieve this, we will use an *upper* bound for the penalty of the tardy jobs and a *lower* bound for the penalty of the early jobs.

Note that in this part of the proof, we do not deal with an *optimal* scheduling of the tardy jobs. Since we only want to calculate a suitable upper bound, it is sufficient to work with the scheduling of the tardy jobs as follows.

We can schedule the tardy jobs of $\mathcal{W}$ in blocks of size $2t$, each consisting of 5 jobs of $\mathcal{W}1$ and one job of $\mathcal{W}2$, as illustrated in Figure 11.

INSERT FIGURE 11 HERE

For each block, this gives a penalty of

$$\begin{aligned} & 6(s_{T_a\,1} - d) + (v-\alpha) + (3v-3\alpha) \\ & +(5v+\alpha+2B) + (4v+5\alpha+3B) + (6v+3\alpha+3B) + (6v+2B) \\ = \quad & 6(s_{T_a\,1} - d) + 25v + 5\alpha + 10B \end{aligned}$$

if T_a is the $\mathcal{T}$-job as in Figure 11.

The (tardy) jobs of $\mathcal{V}$ are scheduled after the jobs of $\mathcal{W}$. Since we cannot assume a solution of 3-Partition in general, we schedule the jobs on M_2 according to the SPT-rule. A sketch can be found in Figure 12.

INSERT FIGURE 12 HERE

An upper bound for the contribution of the jobs of $\mathcal{V}$ scheduled in this way to the objective function value is obtained by the following argumentation. We group the jobs in triples, as done in the NP-hardness proof for $O2|d_i = 0|\sum |C_i - d| + \sum(s_{i\,[2]} - c_{i\,[1]})$. The completion times of the jobs of each triple on M_2 are within the time interval of the processing of one job of $\mathcal{T}$ on M_1. This job of $\mathcal{T}$ is called associated with the triple. The triples are numbered from 1 to n. The first job of the triple is started on M_2 exactly when the associated $\mathcal{T}-$job is started on M_1 (v_1 in Figure 12) or no more than nB earlier (v_4 in Figure 12). Thus, the intermediate storage costs of the first job of a triple i are less than t and the deviation from the due date is less than $s_{\tilde{T}\,1} - d + (i-1)t + v + B$. Using similar arguments for the second job of a triple i, we get as an upper bound for the intermediate storage costs $B + v$ and for the deviation from the due date $s_{\tilde{T}\,1} - d + (i-1)t + 2v + B$. For the third job of the i-th triple, the intermediate storage costs are less than Bn and the deviation from the due date is less than $s_{\tilde{T}\,1} - d + (i-1)t + 3v + B$. Summarizing, we get for the penalty of the jobs of set $\mathcal{V}$:

$$3ns_{\tilde{T}\,1} - 3nd + \sum_{i=0}^{n-1} (3it + 7v + (4+n)B + t) \tag{31}$$

The penalty for these $3n$ jobs is *higher* than the penalty for $3n$ jobs of $\mathcal{W}$ scheduled in blocks as given above. Thus, in order to get a lower bound for the number of tardy jobs of $\mathcal{T}$, we assume that all n jobs of $\mathcal{V}$ are scheduled late. In this case, the upper bound for their contribution to the objective function value is given by Equation (31). The penalty for the early jobs of $\mathcal{T}$ is:

$$\sum_{i=0}^{q-1} ti = PET(q). \tag{32}$$

The early jobs of $\mathcal{W}$ are scheduled according to the LPT-rule:

$$\sum_{i=0}^{\frac{5}{2}q-1} (v-\alpha)i + \sum_{i=0}^{\frac{q}{2}-1} (v + 5\alpha + 2B)i + \frac{5}{4}q^2(v-\alpha) = PEW(q). \tag{33}$$

The tardy jobs of $\mathcal{T}$ contribute with

$$\sum_{i=1}^{\gamma n - q} ti = PTT(q). \tag{34}$$

Equation (31) can be rewritten as

$$3n\left((\gamma-1)n-q\right)t+\sum_{i=0}^{n-1}(3it+7v+(4+n)B+t)=PTV(q). \tag{35}$$

Replacing the sums by the Gaussian formula for sums, we get a differentiable function in q. For the value q^* which minimizes the sum of the described penalties, there holds

$$q^* \leq \left\lceil \frac{8\gamma nt+16v+7B+5\alpha}{16t-15\alpha-5B} \right\rceil. \tag{36}$$

Let $\gamma = 3$ and consider the case $n = 2$. We have to guarantee that $q^* \leq 2n = 4$ for the number of early $\mathcal{T}$-jobs. Since $t > 30\alpha + 10B$, we get

$$\begin{aligned} q^* &\leq \frac{48t+16v+7B+5\alpha}{16t-15\alpha-5B} \\ &< \frac{48t+17v}{15t} < \frac{54t}{15t} < 4. \end{aligned}$$

In the general case of $\gamma \geq 3$ and $n \geq 2$, we have $\frac{1}{2}n < q^* < 2n$. Thus, we can conclude that all schedules with less than n tardy jobs can be transformed into a schedule in which the number of tardy jobs is no less than n by increasing the starting times of all jobs (except the ones of $\mathcal{F}$) by a certain value such that the objective function value is decreased.

5.2.4 Part 2: Necessity of a Solution of 3-Partition for a Subproblem

In this section, we consider schedules with $3k$ tardy jobs of $\mathcal{V}$. Note that $\sum_{i\in\mathcal{V}_\tau} a_i = kB$ due to earlier results. The number q of early jobs of $\mathcal{T}$ is fixed.

We first construct a normal schedule $S^P(k,q)$ under the assumption that a solution of 3-Partition exists for the subproblem with the $3k$ a_i of the tardy $\mathcal{V}$-jobs as input data. We schedule the tardy jobs of $\mathcal{W}$ in blocks as given by Figure 11. The jobs of $\mathcal{V}_\tau$ are scheduled after the last $\mathcal{W}$-job and analogously to the proof of NP-hardness for the case $d = 0$ (see Figure 10). Again, note that the schedule $S^P(k,q)$ defined above is not uniquely determined, but a representative of the set of all possible schedules of this form.

We will only calculate the costs of the jobs of set $\mathcal{V}_\tau \cup \mathcal{W}_\tau$, since the costs of the early jobs are not influenced by the scheduling of the tardy ones. Furthermore, the penalty for the tardy jobs of $\mathcal{T}$ is obviously minimal, since there are no intermediate storage costs. For the penalty of the tardy jobs of $\mathcal{W}$ we get:

$$\frac{1}{2}(25v+5\alpha+10B)(\gamma n-q-k)+12t\sum_{i=0}^{\frac{\gamma n-q-k}{2}-1} i. \tag{37}$$

An upper bound for the penalty for the scheduling of set $\mathcal{V}_\tau$ is

$$3kt(\gamma n - q - k) + \sum_{i=0}^{k-1}(3it + 4v + t + 4B). \tag{38}$$

We will make use of the following definition:

Definition 3 *Two tardy jobs $j, k \in \mathcal{V} \cup \mathcal{W}$ are called an* exact pair *in a schedule S if there is a job $i \in \mathcal{T}$ with $c_{i\,1} = c_{j\,2} = s_{k\,2}$. If one of the equalities is violated, the two tardy jobs are called* inexact pair.

Let us now strengthen a result given in Section 5.2.2. Let $|T_\tau|$ be the number of tardy jobs of $\mathcal{T}$. Denote with

$$I_i = [d + \lambda_{i-1}t, d + \lambda_i t] \tag{39}$$

for $i = 1, \ldots, g$ with $\lambda_0 = 0$, $\lambda_g = |T_\tau|$ time intervals with the property that there are two tardy jobs of sets $\mathcal{V}$ or $\mathcal{W}$, where one finishes on M_2 at $d + \lambda_{i-1}t$ and one at $d + \lambda_i t$. Furthermore, we require that g is maximal which is equivalent to the interval lenghts being minimal. For *all* time intervals I_i, there holds:

- the number of jobs of $\mathcal{W}1$ is a multiple of 5;
- the number of jobs of $\mathcal{W}2$ is exactly one fifth of the number of jobs of $\mathcal{W}1$;
- the number of jobs of $\mathcal{V}$ is a multiple of 3, and the a_i of the jobs processed in I_i sum up to a multiple of B.

It follows that the number

$$ep = \frac{\gamma n - q - k}{2} + k - 1 = \frac{\gamma n - q + k}{2} - 1$$

of exact pairs in schedule S^P is *maximal*. If 3-Partition has no solution, the number of exact pairs is less than ep.

Define

$$\delta_\alpha = \gamma n(5\alpha + 2B). \tag{40}$$

By construction, for the i-th tardy job of $\mathcal{T}$ ($i = 1, \ldots, |T_\tau|$), there are three jobs of $\mathcal{V}$ and $\mathcal{W}$ which are started on M_2 in the intervals

- $[d + t(i-1) - \delta_\alpha, d + t(i-1) + \delta_\alpha]$ (job $i_1 \in \mathcal{V} \cup \mathcal{W}$),
- $[d + t(i-1) - \delta_\alpha + v - \alpha, d + t(i-1) + \delta_\alpha + v + 5\alpha + 2B]$ (job $i_2 \in \mathcal{V} \cup \mathcal{W}$),
- $[d + t(i-1) - \delta_\alpha + 2v - 2\alpha, d + t(i-1) + \delta_\alpha + 2v + 10\alpha + 4B]$ (job $i_3 \in \mathcal{V} \cup \mathcal{W}$).

There are two forms of an inexact pair. First, we can have $s_{(i+1)_1\,2} = d+ti+\epsilon = c_{i_3\,2}$ with $\epsilon > 0$. Then $s_{(i+1)_1\,1} = d + ti$ and $s_{i_3\,1} = d + t(i-1)$. The intermediate storage costs are ϵ for job $(i+1)_1$ and at least $t+\epsilon-v-5\alpha-2B$ for job i_3. The penalty due to the deviation from the due date is at least $ti+\epsilon+v-\alpha$ for job $(i+1)_1$ and $ti+\epsilon$ for job i_3.

The second possibility is $s_{(i+1)_1\,2} = d + ti - \epsilon = c_{i_3\,2}$ with $\epsilon > 0$. We get $s_{(i+1)_1\,1} = d + t(i-1)$ and $s_{i_3\,1} = d + ti$. The intermediate storage costs are $t-\epsilon$ for job $(i+1)_1$ and ϵ for i_3. The penalty due to the deviation from the due date is at least $ti-\epsilon+v-\alpha$ for job $(i+1)_1$ and ti for job i_3.

Thus, if $s_{(i+1)_1\,2} = d + ti = c_{i_3\,2}$, the penalty for the two jobs i_3 and $(i+1)_1$ is more than v less than otherwise.

A lower bound for the best schedule in the absence of a solution of 3-Partition can now be derived as follows. We have at most $ep-1$ exact pairs. Without loss of generality, we schedule these pairs around the first $ep-1$ completion times of the jobs of $\mathcal{T}$. This gives a lower bound for the penalty:

$$\sum_{i=1}^{ep-1}(2it+v-\alpha). \tag{41}$$

Since the penalty for $s_{(i+1)_1\,2} = d+ti+\epsilon = c_{i_3\,2}$ with $0 < \epsilon < \delta_\alpha$ is lower than the second possibility of scheduling inexact pairs, we get as a lower bound for their contribution to the objective function value:

$$\sum_{i=ep}^{\gamma n-q-1}(\epsilon+t+\epsilon-v-5\alpha-2B+ti+\epsilon+v-\alpha+ti+\epsilon) > \sum_{i=ep}^{\gamma n-q-1}(t+2ti-6\alpha-2B). \tag{42}$$

A lower bound for the sum of the penalties of the very first and the last tardy job of $\mathcal{V}_\tau \cup \mathcal{W}_\tau$ is:

$$v-\alpha+(\gamma n-q)t. \tag{43}$$

The costs of the middle job of a triple is at least

$$\sum_{i=0}^{\gamma n-q-1}(it+3v-3\delta_\alpha). \tag{44}$$

Now, we add the terms in (37) and (38) and subtract the terms (41) to (44). We get

$$\begin{aligned}
&-2v+B(3\gamma n-3q+k+1)+3\delta_\alpha(\gamma n-q)+\alpha(6\gamma n-6q-5k-10)\\
&<-2v+4\gamma nB+18\gamma^2n^2+6\alpha\gamma n\\
&<-2v+\alpha(4+24\alpha)<-2v+28\alpha^2<-v<-\Delta.
\end{aligned} \tag{45}$$

This shows that the objective function value of every schedule S^P basing on a solution of 3-Partition of a subproblem with $3k$ elements is more than Δ lower than the objective function value of any other schedule with $3k$ tardy jobs of $\mathcal{V}$ and the same number q of early jobs of $\mathcal{T}$.

To conclude this section, let us point out that the decision to schedule the jobs of set $\mathcal{V}$ after the last job of $\mathcal{W}$ has been made for the sake of convenience only. It is easy to see that there holds the following property. Consider a normal schedule with time intervals I_i $(i = 1, \ldots, g)$ as defined above and g partial sequences π_i describing the order of processing of the jobs of sets $\mathcal{V}$ and $\mathcal{W}$ in the time interval I_i. Let ν be a permutation of the numbers $1, \ldots, g$. Then changing the sequence $(\pi_1, \pi_2, \ldots, \pi_g)$ for the tardy jobs of $\mathcal{V}$ and $\mathcal{W}$ of this schedule into the sequence $(\pi_{\nu(1)}, \pi_{\nu(2)} \ldots, \pi_{\nu(g)})$ will not change the objective function value.

5.2.5 Part 3: No More Than 3 Jobs of $\mathcal{V}$ are Scheduled Early

Consider a schedule with $3k$ tardy jobs of $\mathcal{V}$. Let q be the number of early jobs of $\mathcal{T}$. So far, we have shown that a lower bound for this schedule for given q and k is $LB(k, q)$ as defined in Section 5.2.1. As mentioned at the end of Section 5.2.2, the early jobs of $\mathcal{V} \cup \mathcal{W}$ are scheduled according to the LPT-rule on M_2 in a normal schedule. Thus, at first the jobs of $\mathcal{W}2$ are processed on M_2, then the jobs of $\mathcal{V}$ and then the $\mathcal{W}1$-jobs. Immediately after the due date the jobs of $\mathcal{W}_\tau$ are scheduled in blocks as given in Figure 11 on M_2 and finally the jobs of $\mathcal{V}_\tau$ as described in Figure 10.

We assume that a solution of 3-Partition for the complete instance with $3n$ values a_i of the form given in Equations (13) and (14) exists. We will write $S^P(k, q)$ for a schedule with $3k$ tardy jobs of $\mathcal{V}$, which bases on a solution of 3-Partition for the subproblem with the $3k$ elements

$$\{a_1, a_2, \ldots, a_k, a_{1+n}, a_{2+n}, \ldots, a_{k+n}, a_{1+2n}, a_{2+2n}, \ldots, a_{k+2n}\}$$

of these tardy jobs as input data. This simplifies the presentation, since the schedule $S^P(k+2, q)$ can be constructed easily out of $S^P(k, q)$ by, among other operations, scheduling additionally the six $\mathcal{V}$-jobs $(0, v + a_{k+1})$, $(0, v + a_{k+1+n})$, $(0, v + a_{k+1+2n})$, $(0, v + a_{k+2})$, $(0, v + a_{k+2+n})$, $(0, v + a_{k+2+2n})$ tardy. In general, it is not possible to extend an arbitrary solution of a subproblem of 3-Partition with $3k$ elements to a solution of the problem with $3(k+2)$ elements in the way described above.

We have to prove that if a solution of 3-Partition exists, there is a schedule where no more than 3 jobs of $\mathcal{V}$ are scheduled early. We show this inductively by proving that under the assumption that if a solution of 3-Partition (for the a_i of the complete problem with $3n$ $\mathcal{V}$-jobs) exists, there is a schedule $S^P(k+2, q)$ with $3(k+2)$ tardy $\mathcal{V}$-jobs with an objective function value which is at least Δ less than $LB(k, q)$.

We will now develop an upper bound for the objective function value of schedule $S^P(k+2,q)$ through comparison with schedule $S^P(k,q)$. We do not alter the starting times of the jobs of $\mathcal{W}$ which are already early in $S^P(k,q)$. Denote with h the starting time on M_2 of the job of set $\mathcal{W}1$ which is processed first in schedule $S^P(k,q)$. The starting times on M_2 of the jobs of $\mathcal{V}_\epsilon$ of schedule $S^P(k+2,q)$ are chosen such that the jobs are processed (without idle time) in the time interval $[h-(n-k-2)t, h]$. The six jobs of $\mathcal{W}$ which are scheduled tardy in $S^P(k,q)$, but early in $S^P(k+2,q)$ are processed in the latter schedule in the time interval $[h-(n-k)t, h-(n-k-2)t]$ on M_2. (Note that this is a slight deviation from a normal schedule to simplify the presentation — recall that we are calculating an upper bound.) We schedule the six additional tardy jobs of $\mathcal{V}$ in place of the six jobs of $\mathcal{W}$ which are processed last in $S^P(k,q)$. Apart from this, we maintain the order of processing of the jobs of set $\mathcal{V}$ which remain early in schedule $S^P(k+2,q)$.

An upper bound for the penalty for the six jobs of $\mathcal{V}$ which are processed early in $S^P(k,q)$, but tardy in $S^P(k+2,q)$ is in schedule $S^P(k+2,q)$ (see Equations (15) to (17) with $n=2$):

$$\begin{aligned} &6t(\gamma n - q - k - 2) + (t + 2v + B) + (t + 6v + 3B) + (3t) = \\ &6t(\gamma n - q - k - 2) + 5t + 8v + 4B, \end{aligned} \tag{46}$$

wheras a lower bound for their contribution to the objective function value of $S^P(k,q)$ is

$$6\left(\frac{5}{2}(q-n+k)(v-\alpha) + (n-k)t\right) - \sum_{i=1}^{6} i(v+B). \tag{47}$$

The penalty for the six jobs of $\mathcal{W}$ (five of $\mathcal{W}1$ and one of $\mathcal{W}2$) which are the last ones scheduled tardy in $S^P(k,q)$ is

$$25v + 5\alpha + 10B + (\gamma n - q - k - 2)6t. \tag{48}$$

The costs of scheduling them early in $S^P(k+2,q)$ are

$$6\left(\frac{5}{2}(q-n+k)(v-\alpha) + (n-k)t\right) - 6(v + 2B + 5\alpha) - \sum_{i=1}^{5} i(v-\alpha) \tag{49}$$

Adding the terms given in (46) and (49) and subtracting the terms in (47) and (48), we get

$$-2v + 8B - 20\alpha < -\Delta \tag{50}$$

which proves that in a schedule with an objective function value which is less than $Z + \Delta$ the number $|\mathcal{V}_\epsilon|$ of early $\mathcal{V}$-jobs is no greater than 3.

5.2.6 Part 4: Sufficiency of a Solution of 3-Partition

It remains to be shown that a schedule based on a solution for the 3-Partition problem (with $3n$ a_i) has an objective function value less than $Z + \Delta$. To see this, note that an *upper* bound for any schedule of the described form is derived from $LB(k, q)$ by adding

$$\sum_{i=0}^{3(n-k-1)} iB \tag{51}$$

to the lower bound for the penalty of the jobs of $\mathcal{V}_\epsilon$, and $8Bk$ to the penalty for $\mathcal{V}_\tau$. The sum of both terms is less than Δ. ∎

6 Conclusions and Further Work

This article has studied various possibilities to generalize one-machine problems with a non-regular perfomance measure to a multi-stage environment. The computational complexities of these problems can be summarized in Table 4:

Problem	Due Date	Complexity	Section
$J\mu \vert\vert \sum \vert C_i - d \vert$	non-restrictive	polynomial	2.3
$F\mu \vert\vert \sum \vert C_i - d \vert$	non-restrictive	polynomial	2.3
$O\mu \vert\vert \sum \vert C_i - d \vert$	non-restrictive	polynomial	2.3
$F2 \vert\vert \sum \vert C_i - d \vert + \sum (s_{i\,2} - c_{i\,1})$	$d = 0$	unary NP-hard	4
$F2 \vert\vert \sum \vert C_i - d \vert + \sum (s_{i\,2} - c_{i\,1})$	non-restrictive	unary NP-hard	4
$O2 \vert\vert \sum \vert C_i - d \vert + \sum (s_{i\,[2]} - c_{i\,[1]})$	$d = 0$	unary NP-hard	5.1
$O2 \vert\vert \sum \vert C_i - d \vert + \sum (s_{i\,[2]} - c_{i\,[1]})$	non-restrictive	unary NP-hard	5.2

Table 4: Summary of the complexities of the problems considered in this article

We want to conclude this article by giving some suggestions for further resarch on this topic. The enumerative algorithm presented in Section 3.2 can be developed further to a branch and bound algorithm capable to solve instances with more than ten jobs in a reasonable time. Additionally, this algoritm can be generalized to non-restrictive due dates. Though, from our own experience with developing a branch and bound algorithm for a problem without intermediate storage costs,[5] we expect that this will be a challenging task. Moreover, the average problem size which can be solved on a PC in some minutes will be significantly lower than 20. Therefore, developing heuristics for the problems presented here will be a commendable contribution to the applicability of scheduling theory to real world problems.

References

[1] J. Kanet. Minimizing the average deviation of job completion times about a common due date. *Naval Res. Logist. Quart.*, 28:643–651, 1981.

[2] N. Hall and M. Posner. Earliness-tardiness scheduling problems, i: Weighted deviation of completion times about a common due date. *Oper. Res.*, 5:836–846, 1991.

[3] H. Sarper. Minimizing the sum of absolute deviations about a common due date for the two-machine flow shop problem. *Appl. Math. Modelling*, 19:153–161, 1995.

[4] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling. *Annals of Disc. Math.*, 5:287–326, 1979.

[5] J. N. D. Gupta, V. Lauff, and F. Werner. A branch and bound algorithm for two-machine flow shop problems with earliness and tardiness penalties. *Otto-von-Guericke-Universität, FMA, Preprint 33/99*, 1999.

[6] W. Szwarc. Single machine scheduling to minimize absolute deviation of completion times from a common due date. *Naval Res. Logist. Quart.*, 36:663–673, 1989.

[7] U. Bagchi, R. S. Sullivan, and Y. L. Chang. Minimizing mean absolute deviation of completion times about a common due date. *Naval Res. Logist. Quart.*, 33:227–240, 1986.

[8] K. R. Baker and G. D. Scudder. Sequencing with earliness and tardiness penalties. *Oper. Res.*, 38:22–36, 1989.

[9] S. T. Webster. The complexity of scheduling job families about a common due date. *Oper. Res. Lett.*, 20:65–74, 1997.

[10] M. R. Garey, D. S. Johnson, and R. Sethi. The complexity of flowshop and jobshop scheduling. *Math. Oper. Res.*, 1:117–129, 1976.

[11] W. Kubiak, S. Lou, and S. Sethi. Equivalence of mean flow time problems and mean absolute deviation problems. *Oper. Res. Lett.*, 9(6):371–374, 1990.

[12] J. N. D. Gupta. Optimal flowshop schedules with no intermediate storage space. *Naval Res. Logist. Quart.*, 23:235–243, 1976.

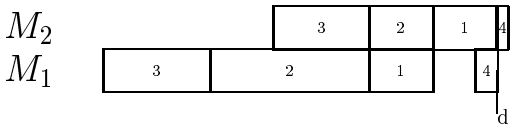

Figure 1: A best permutation schedule for Example 3

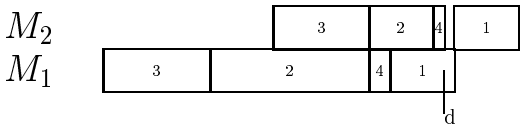

Figure 2: An alternative best permutation schedule for Example 3

Figure 3: Optimal schedule for Example 3

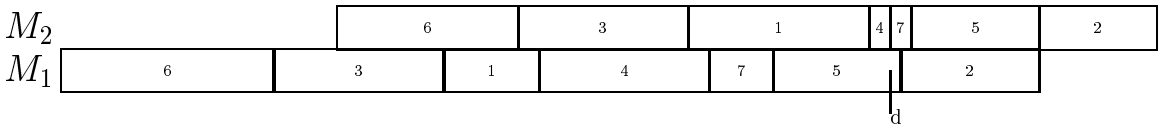

Figure 4: Schedule S^0

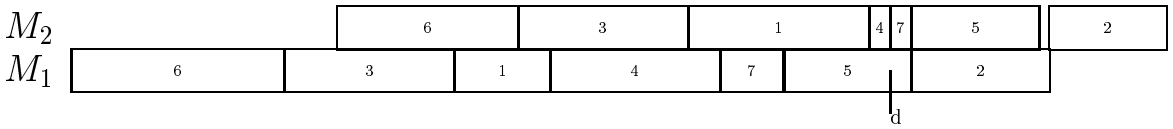

Figure 5: Schedule S^1

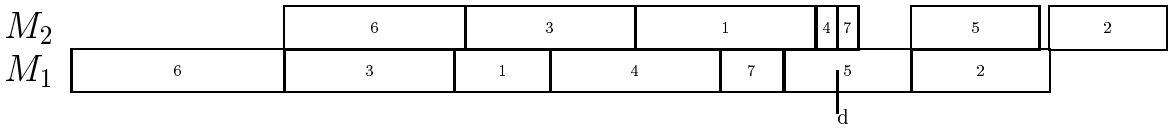

Figure 6: Schedule S^2

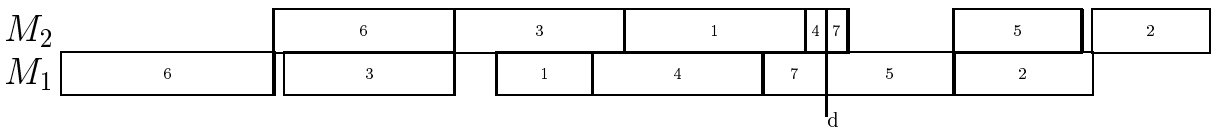

Figure 7: Schedule S^3

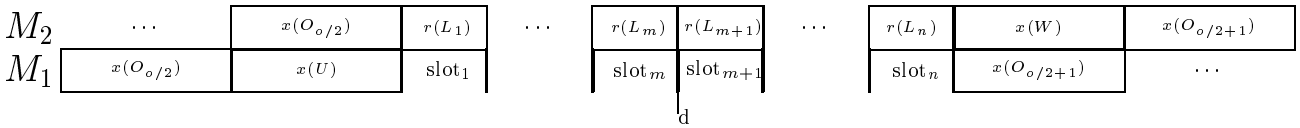

Figure 8: Skeleton schedule

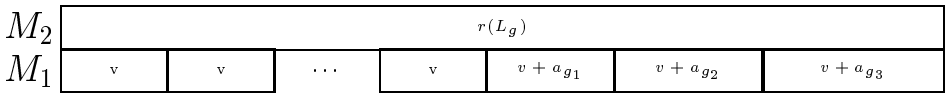

Figure 9: A slot

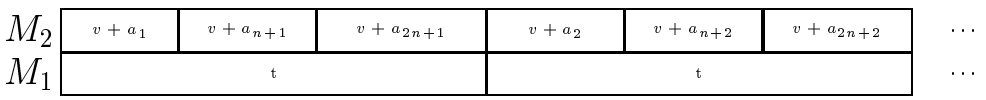

Figure 10: Schedule for $O2|d_i = 0|\sum |C_i - d| + \sum (s_{i\,[2]} - c_{i\,[1]})$

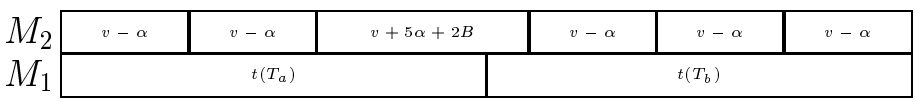

Figure 11: Block of six tardy jobs of $\mathcal{W}$

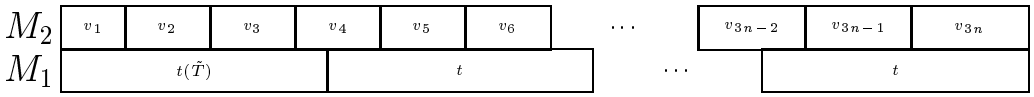

Figure 12: Tardy jobs of $\mathcal{V}$