INT. J. PROD. RES., 15 JUNE 2004, VOL. 42, NO. 12, 2435–2456

Taylor & Francis
Taylor & Francis Group

Single-track multi-hoist scheduling problem: a collision-free resolution based on a branch-and-bound approach

A. CHE and C. CHU*

An analytical mathematical model and a branch-and-bound algorithm for single-track cyclic multi-hoist scheduling problems are proposed. The objective is to minimize the cycle time for a given number of hoists. The collision-free single-track constraints are first formulated as disjunctive inequalities. It is then shown that this formulation is a very strict and necessary condition. To be a sufficient and necessary one, two additional properties, like collision-checking rules, must hold in optimal solutions. It is proved that a solution violating these two properties due to their relaxation is always dominated by a collision-free one. Therefore, these two properties are relaxed in the branch-and-bound algorithm. The computation of lower bounds in the branch-and-bound algorithm requires the solution of a specific linear programming problem, which can be solved by using a graph-based polynomial algorithm. Computational results with both benchmark and randomly generated test instances are presented.

1. Introduction

Highly automated or even unmanned production lines where material handling is performed by computer-controlled robots or hoists have become commonplace in contemporary manufacturing systems. In such systems, the hoists or robots are in charge of transporting parts from one machine to another for the next operation. The productivity of these systems largely depends on the schedule of the robot activities, especially when the intermediate storage area is limited in order to limit the work-in-process. Due to the importance of the problem, a lot of work addressing production systems involving robots or hoists for material handling has been published in recent years.

1.1. *Motivation of the study*

The application that motivated this study is an automated electroplating line for processing printed circuit boards (PCBs). Such a production line is composed of a sequence of chemical tanks and a crew of identical programmable hoists (figure 1). Each tank contains chemicals required for a specific electroplating step in the processing of parts, such as acid cleaning, acid activating, copper plating, rinsing, etc. Due to the characteristics of chemical processes in PCB electroplating or other galvanization processes, the processing time of a part in each tank is usually bounded both from below and from above. This constraint is usually called a time-window constraint. The movements of parts between the tanks are performed by a crew of

Revision received June 2003.

Laboratoire d'Optimisation des Systèmes Industriels (LOSI), Université de Technologie de Troyes, 12 Rue Marie Curie, BP 2060, F-10010, Troyes, Cedex, France.

*To whom correspondence should be addressed. e-mail: Chengbin.chu@utt.fr

International Journal of Production Research ISSN 0020–7543 print/ISSN 1366–588X online
http://www.tandf.co.uk/journals
DOI: 10.1080/00207540410001666288

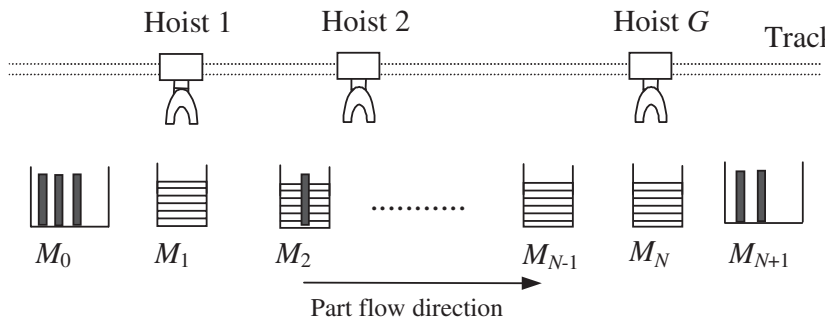


Figure 1. Layout of an automated electroplating line for processing PCBs.

computer-controlled hoists on a single track. Effective scheduling of hoist movements is therefore critical to achieving high productivity. This problem is commonly known as the hoist scheduling problem (Phillips and Unger 1976, Shapiro and Nuttle 1988, Song *et al*. 1993, Lei and Wang 1994, Sun *et al*. 1994, Varnier *et al*. 1997, Chen *et al*. 1998, Kats *et al*. 1999, Che *et al*. 2002, 2003).

In most real industrial environments, tanks are arranged in a row. The production line can be uni- or multidirectional according to the part flow direction. In a unidirectional production line, the processing sequence of parts is the same as the sequence of tank arrangement, i.e. the part flow is unidirectional. In real production systems, parts may also change its flow direction in some tanks not only in order to minimize the identical tanks in the system, but also because the loading station and the unloading station are often associated (Sun *et al*. 1994, Varnier *et al*. 1997). Thus, transfer of parts between the tanks may also be multi-directional. Such a production line is accordingly called a multidirectional one.

1.2. *Cyclic hoist scheduling problem*

Depending on the control organization of a production line, the hoist-scheduling problem (HSP) can be distinguished into cyclic HSP, real-time HSP, predictive HSP and dynamic HSP. For detailed definitions of different HSPs, see Bloch and Manier (1999).

The present paper considers the cyclic hoist-scheduling problem. A cyclic production system periodically repeats the same state. The hoists perform a fixed sequence of moves repeatedly. Each repetition of the sequence of moves is called a cycle. The duration of a cycle is the cycle time. The cycle time measures the throughput rate of a production system. The fixed sequence of moves that the hoists perform during each cycle is defined by a cyclic schedule. This paper considers simple cyclic schedules. With a simple cyclic schedule, exactly one part is introduced in and removed from each tank during a cycle.

1.3. *Literature review*

Many researchers have studied the cyclic hoist scheduling problem with a single hoist (Phillips and Unger 1976, Shapiro and Nuttle 1988, Song *et al*. 1993, Lei and Wang 1994, Chen *et al*. 1998, Kats *et al*. 1999, Che *et al*. 2002, 2003). The cyclic single-hoist scheduling with time-window constraints has been proved to be NP-hard (Lei and Wang 1989). Scheduling of multiple hoists on a single track further complicates the problem and makes it extremely difficult to find the global optimal

solution. Due to the complexity of the problem, little research work has been done in this area.

Karzanov and Livshits (1978) appear to be the first authors to study the cyclic multi-hoist-scheduling problem. They studied the system with parallel tracks and constant processing times and proposed an $O(N^3)$ algorithm to find the minimal number of robots for a given cycle time, where N is the number of machines in a production line. Kats and Levner (1997) extended their results and found that the problem of minimizing the number of robots for all possible cycle times can be solved in $O(N^5)$ time complexity.

In cyclic scheduling of multi-hoist with time window and single-track constraints, Lei and Wang (1991) proposed a heuristic algorithm, called minimum common cycle, for cyclic two-hoist scheduling problems. To avoid hoist collisions in their movements, their algorithm uses a partitioning approach by which a given production line is partitioned into two no-overlapping zones and each hoist is assigned to a zone. Lei *et al*. (1993) and Armstrong *et al*. (1996) extended this algorithm to multi-hoist cases and studied the problem of minimizing the number of hoists for a given cycle time.

Varnier *et al*. (1997) also developed a two-step approach for the single-track cyclic multi-hoist-scheduling problem in a multi-directional production line. They first used some heuristic rules to partition the production line into overlapping zones. An optimal cyclic schedule is then found for the given hoist assignment using constraint logic programming.

Lamothe *et al*. (1996) presented a multi-hoist model for the dynamic hoist-scheduling problem in a multi-directional line. In their model, each hoist has a safe zone where no other ones can come, but any two adjacent hoists may have an overlapping zone. With this scheme, the multi-hoist conflict problem was reduced to a two-hoist conflict problem. The constraint for avoiding a two-hoist conflict was formulated as a disjunctive constraint.

Similar collision problems can also be found in automated guided vehicle (AGV) scheduling in flexible manufacturing systems, where AGVs travel on a network of guidepath. There are two kinds of commonly used network configurations: single loop and multiloop. In the former, the avoidance of AGV collision is relatively easy to address. In the latter, AGV collisions become major concerns in scheduling (Co and Tanchoco 1991, Lei *et al*. 1997).

1.4. *Outline of the proposed collision-free resolution approach*

Previous approaches toward solving the collision-free constraints have been limited to partitioning a given production line into non-overlapping or overlapping zones using some heuristic rules and each hoist is assigned to a zone. It is apparent that they cannot guarantee the optimality of the obtained solution. This paper proposes a new approach to dealing with the collision-free constraints.

To avoid possible hoist collisions between any two hoist movements A and B using a common segment of the track, either A must precede B or B must precede A. Therefore, in this research, the collision-free constraints are formulated as disjunctive inequalities. We show that this formulation is a necessary condition. To be a sufficient and necessary one, two additional properties, like collision-checking rules, must hold in any feasible solution. We prove that a solution violating these two properties due to relaxation of them is always dominated by a collision-free one.

Consequently, these two properties are relaxed in the proposed branch-and-bound algorithm.

The paper is organized as follows. The single-track cyclic multi-hoist-scheduling problem is formulated in section 2. Section 3 discusses some special properties in the collision-free constraints. A branch-and-bound algorithm is presented in section 4 based on the proposed model. Section 5 gives computational results and section 6 is the conclusion.

2. Problem definition and formulation

2.1. *The problem*

The production line considered is composed of N tanks, $M_1, \ldots, M_N$ and G identical hoists on a single track (figure 1). Two dummy tanks M_0 and M_{N+1} represent the loading and unloading stations, respectively. The part flow can be described as follows. After a part is removed from M_0, it is processed successively through tanks $M_1, M_2, \ldots, M_N$, and leaves the system from M_{N+1}. With this part flow scheme, the processing sequence of parts is the same as the sequence of tank arrangement. G identical hoists on a single shared track are responsible for moving parts between the tanks. For simplicity, the hoist movement of transporting a part from M_i to M_{i+1} is called move i, $0 \leq i \leq N$. Each move i consists of three simple hoist operations: (1) unload a part from M_i; (2) transport the part to the next tank M_{i+1}; and (3) load the part into M_{i+1}. The hoist movement without transporting parts is accordingly called a void move.

The hoists are programmed to perform a fixed sequence of moves repeatedly. The sequence of moves performed by the hoists during each cycle is represented by a cyclic schedule P. A schedule P with multiple hoists on a single track is feasible if and only if it satisfies the following four families of constraints:

- Time window constraints: processing time of a part in a tank M_i must fall into its time window, $i = 1, 2, \ldots, N$.
- Tank capacity constraints: tank M_i, $i = 1, 2, \ldots, N$, must be empty when a part arrives at M_i.
- Hoist capacity constraints: there must be sufficient time for any hoist to travel between the successive moves assigned to that hoist.
- Collision-free constraints: hoists travel along a single shared track and no hoist collisions happen during the cycle.

2.2. *Mathematical model*

2.2.1. *Hypothesis*

This paper assumes that the production line is unidirectional, i.e. the processing sequence of parts is exactly the same as the sequence of tank arrangement. A single type of part is to be processed in this production line. Each tank can process only one part at a time. It is further assumed at any time of a cycle that any hoist is performing either a loaded move or a void move from a tank (to which it has just loaded a part) to another tank (from which it will unload a part), or waiting on a tank for completion of a part. However, the hoists are allowed to pause, if necessary, during performance of this void move in order to avoid possible collisions.

2.2.2. *Notation*

To formulate the problem considered here, the following notation is defined.

- N number of tanks in the production line, not including the loading station and the unloading station,
- G number of hoists on the track. Without loss of generality, it is assumed that the hoists are numbered in the same order as the tanks in the production line (figure 1),
- a_i minimum processing time of parts in M_i, $i = 1, \ldots, N$,
- b_i maximum processing time of parts in M_i, $i = 1, \ldots, N$,
- θ_i time required for the hoist to execute move i, $i = 0, \ldots, N$,
- ε_i time required to unload a part from M_i, $i = 0, \ldots, N$,
- τ_i time required to load a part to M_i, $i = 0, \ldots, N$,
- $\delta_{i,j}$ time required for the empty hoist to move from M_i to M_j, $i, j = 1, \ldots, N+1$,
- β allowed minimum interval between two adjacent hoists on the track to avoid collision. For simplicity, β is measured in time and is equal to the width of the hoists divided by the travelling speed of the hoists.

The following variables are used in the development of the mathematical model:

- x cycle time,
- s_i starting time of move i in a cycle, $i = 0, \ldots, N$,
- c_i 0–1 variable. $c_i = 0$, M_i contains no part at the beginning of a cycle; $c_i = 1$ M_i contains a part at the beginning of a cycle, $i = 0, \ldots, N$. Note that c_0 is always equal to 1 since it is assumed that a part is always available at the loading station,
- r_i index of the hoist performing move i, $r_i \in \{1, 2, \ldots, G\}$, $i = 0, \ldots, N$,
- S set of pairs of moves performed by a same hoist, i.e. $S = \{(i,j) | r_i = r_j, i, j = 0, 1, \ldots, N\}$,
- D set of pairs of moves performed by two hoists using a common segment of the track, i.e. $D = \{(i,j) | i < j, r_i > r_j \text{ or } i > j, r_i < r_j, i, j = 0, 1, \ldots, N\}$,
- Q set of pairs of moves performed by two hoists but not using a common segment of the track, i.e. $Q = \{(i,j) | i < j, r_i < r_j \text{ or } i > j, r_i > r_j, i, j = 0, 1, \ldots, N\}$, $C = \{c_0, c_1, \ldots, c_N\}$, where C is the initial part distribution of a cycle, $R = \{r_0, r_1, \ldots, r_N\}$, where $r_i \in \{1, 2, \ldots, G\}$, $0 \le i \le N$, and R is the hoist distribution of a cycle, $z_i = (c_i, r_i)$, where z_i is said to be the state of tank M_i, $i = 0, \ldots, N$, $Z = \{z_0, z_1, \ldots, z_N\}$, where Z is the tank state distribution.

By definition of a move:

$$\theta_j \ge \delta_{j,j+1}, \quad \text{for all } j = 0, 1, \ldots, N. \tag{1}$$

The present paper makes the following assumption commonly known as the triangle inequality:

$$\delta_{i,j} \le \delta_{i,k} + \delta_{k,j}, \quad \text{for all } i, j, k = 0, 1, \ldots, N. \tag{2}$$

2.2.3. *Formulation of the time window constraints*

The time window constraints ensure that a part is processed in M_i, $1 \le i \le N$, for a period no less than a_i and no more than b_i. Violation of the time window constraints

would cause a defective part. The time window constraints can be formulated as (Chen *et al*. 1998):

$$a_i \le s_i + c_i x - s_{i-1} - \theta_{i-1} \le b_i, \quad \text{for all } i = 1, 2, \ldots, N. \tag{3}$$

2.2.4. *Formulation of the hoist capacity constraints*

The hoist capacity constraints ensure that any hoist is scheduled to handle only one part at a time. Let $\sigma_k = < l_0^k l_1^k \cdots l_{N_k-1}^k l_{N_k}^k >$, $1 \le k \le G$, be the sequence of moves for hoist k corresponding to a cyclic schedule, where N_k is the number of moves performed by hoist k. Then the hoist capacity constraints can be described as follows:

$$s_{l_i^k} + \theta_{l_i^k} + \delta_{l_i^k+1, l_{i+1}^k} \le s_{l_{i+1}^k}, \quad \text{for all } i = 0, \ldots, N_k - 1, \quad \text{for all } k = 1, 2, \ldots, G \tag{4}$$

$$s_{l_{N_k}^k} + \theta_{l_{N_k}^k} + \delta_{l_{N_k}^k+1, l_0^k} \le x + s_{l_0^k}, \quad \text{for all } k = 1, 2, \ldots, G. \tag{5}$$

Relation (4) states that on completion of move l_i^k, hoist k has sufficient time to travel from tanks $l_i^k + 1$ to l_{i+1}^k. Relation (5) means that hoist k has sufficient time to travel to the tank corresponding to its first move in the next cycle after completing its last move. By considering (1) and (2), constraint (4) can be equivalently expressed as:

$$s_i + \theta_i + \delta_{i+1,j} \le s_j, \quad \text{for all } s_i < s_j, (i,j) \in S. \tag{6}$$

In view of (1) and (2), (5) can be rewritten as:

$$s_j + \theta_j + \delta_{j+1,i} \le x + s_i, \quad \text{for all } s_i < s_j, (i,j) \in S. \tag{7}$$

Considering (6) and (7):

$$s_i + \theta_i + \delta_{i+1,j} \le x + s_j, \quad \text{for all } (i,j) \in S \tag{8}$$

$$s_j + \theta_j + \delta_{j+1,i} \le x + s_i, \quad \text{for all } (i,j) \in S. \tag{9}$$

2.2.5. *Formulation of the collision-free constraints*

As mentioned above, the present paper considers a unidirectional production line, where the processing sequence of parts is the same as the sequence of tank arrangement. For any two hoists k_1 and k_2 such that $k_1 < k_2$, hoist k_1 is before hoist k_2; otherwise, hoist k_1 is after hoist k_2. Similarly, for any two tanks M_i and M_j such that $i < j$, M_i is before M_j; otherwise, M_i is after M_j.

For any two hoists k_1 and k_2 such that $k_1 < k_2$, if hoist k_1 is always before hoist k_2 in the production line at any time of a cycle, no collisions would happen during the cycle and the collision-free constraints are satisfied. With the unidirectional part flow scheme, it is apparent that no collisions will happen between two hoists during their performance of loaded moves. Collisions may happen between two hoists using a common segment of the track only when one hoist is performing a void move and another is either performing a move, or a void move, or waiting for the completion of a part. It should be noted that collisions may also happen between two hoists sharing a same tank, where parts are loaded/unloaded by one hoist and unloaded/loaded by another one. Such a tank is called a boundary tank in this paper. Therefore, two types of collisions have to be dealt with in the formulation of the collision-free constraints.

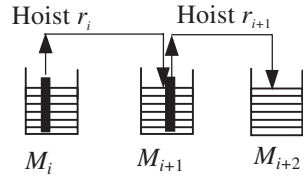


Figure 2. Two hoists sharing a boundary tank.

2.2.5.1. *Two hoists sharing a boundary tank*. For two moves i and $i+1$, $1 \le i \le N-1$, it is first assumed that $r_i < r_{i+1}$. In this case, hoists r_i and r_{i+1} may collide at M_{i+1} (figure 2), where parts are loaded by hoist r_i and unloaded by hoist r_{i+1}. Two cases should be considered here.

(1) Hoist r_{i+1} unloads a part from M_{i+1} before hoist r_i loads another part into M_{i+1}, i.e. $c_{i+1} = 1$. Hoist r_{i+1} performing move $i+1$ leaves M_{i+1} at time $s_{i+1} + \varepsilon_{i+1}$, and hoist r_i performing move i arrives at M_{i+1} at time $s_i + \theta_i - \tau_{i+1}$. Between the time hoist r_{i+1} leaves M_{i+1} and the time hoist r_i arrives at M_{i+1}, other $(r_{i+1} - r_i)$ hoists before hoist r_{i+1} and after hoist r_i must also pass over M_{i+1}. Therefore, in order to avoid collision, the following condition must hold:

$$s_{i+1} + \varepsilon_{i+1} + (r_{i+1} - r_i)\beta \le s_i + \theta_i - \tau_{i+1}, \quad \text{for all } c_{i+1} = 1, (i+1, i) \in Q.$$

This relation can be rewritten as:

$$s_{i+1} \le s_i + \theta_i - \varepsilon_{i+1} - \tau_{i+1} - (r_{i+1} - r_i)\beta, \quad \text{for all } c_{i+1} = 1, (i+1, i) \in Q. \tag{10}$$

(2) Hoist r_{i+1} unloads a part from M_{i+1} after hoist r_i loads that part into M_{i+1}. Similarly, we obtain:

$$s_i + \theta_i - \varepsilon_{i+1} - \tau_{i+1} + (r_{i+1} - r_i)\beta \le s_{i+1}, \quad \text{for all } c_{i+1} = 0, (i+1, i) \in Q. \tag{11}$$

Due to the time window constraints, we have:

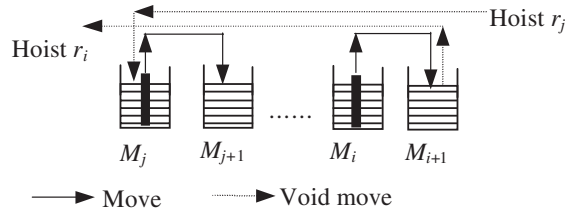
$$s_i + \theta_i + a_i \le s_{i+1}.$$

Since $\theta_i + a_i \gg \theta_i - \varepsilon_{i+1} - \tau_{i+1} + (r_{i+1} - r_i)\beta$, inequality (11) is redundant with the consideration of the time window constraints.

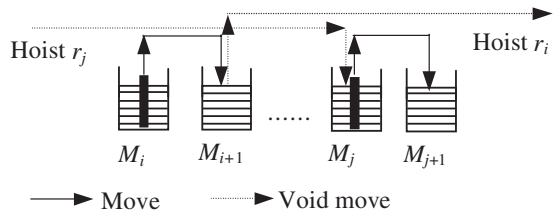
If $r_i > r_{i+1}$, hoists r_i and r_{i+1} share not only the boundary tank M_{i+1}, but also the segment of the track from M_i to M_{i+2}. If no collisions happen between them in using the segment of the track from M_i to M_{i+2}, then there must be no collisions between them in using the boundary tank M_{i+1}. As a result, these constraints are redundant with the consideration of the constraints for using the common segment of the track between these two hoists, and therefore need not to be considered here.

2.2.5.2. *Two hoists using a common segment of the track*. In the following, for any two moves i and j, without loss of generality, it is assumed that $s_i < s_j$. If $i > j$ and $r_i < r_j$, or $i < j$ and $r_i > r_j$, then hoists r_i and r_j will share a segment of the track (figures 3a, b).

To avoid the collision, after hoist r_i finishes performing move i, it should arrive at M_j before hoist r_j (figures 3a, b). Note that the earliest time hoist r_i arrives at M_j is $s_i + \theta_i + \delta_{i+1,j}$, and the latest time hoist r_j arrives at M_j is s_j. Between the time hoist



(a)



(b)

Figure 3. (a) Two hoists using a common zone of the track ($i > j$, $r_i < r_j$); (b) Two hoists using a common zone of the track ($i < j$, $r_i > r_j$).

r_i arrives at M_j and the time hoist r_j arrives at M_j, other $|r_i - r_j|$ hoists must also pass over M_j. Therefore, the following constraint must be satisfied:

$$s_i + \theta_i + \delta_{i+1,j} + |r_i - r_j|\beta \le s_j, \quad \text{for all } s_i < s_j,\ (i,j) \in D. \tag{12}$$

In addition, the following relation must also hold:

$$s_j + \theta_j + \delta_{j+1,i} + |r_i - r_j|\beta \le x + s_i, \quad \text{for all } s_i < s_j,\ (i,j) \in D. \tag{13}$$

Relation (13) says that hoist r_j performing move j must pass over M_i before hoists r_i arrives at M_i to perform move i in the next cycle.

It follows from (12) that:

$$s_i + \theta_i + \delta_{i+1,j} + |r_i - r_j|\beta \le x + s_j, \quad \text{for all } s_i < s_j,\ (i,j) \in D. \tag{14}$$

By taking into account the redundant constraint (14), (13) can be equivalently written as:

$$s_j + \theta_j + \delta_{j+1,i} + |r_i - r_j|\beta \le x + s_i, \quad \text{for all } s_i < s_j,\ (i,j) \in D \tag{15}$$

$$s_i + \theta_i + \delta_{i+1,j} + |r_i - r_j|\beta \le x + s_j, \quad \text{for all } s_i < s_j,\ (i,j) \in D. \tag{16}$$

The inconsistency problem that may be caused by this formulation of the collision-free constraints will be discussed below.

Note that (12) and (13) are necessary conditions for avoiding possible collisions between any two hoists using a common segment of the track. To illustrate this observation, consider the following example.

Example 1: A system is composed of 12 tanks. The data for this example are shown in table 1. For any i and j such that $0 \le i < j \le N+1$, the time for a void move from M_i to M_j is obtained with $\delta_{i,j} = \delta_{j,i} = \sum_{k=i}^{j-1} \delta_{k,k+1}$. For any i such that $0 \le i \le N$, $\varepsilon_i = 8.5$, $\tau_i = 11.5$, $\theta_i = \delta_{i,i+1} + 20$. Set $\beta = 0.5$. Table 2 gives the optimal solution

i	0	1	2	3	4	5	6	7	8	9	10	11	12
a_i	0	56	56	143	127	137	148	112	86	144	72	79	128
b_i	$+\infty$	68	86	146	140	184	193	133	129	164	78	110	140
$\delta_{i,i+1}$	2	2	2	2	2	1	2	1	1	1	1	2	1
θ_i	22	22	22	22	22	21	22	21	21	21	21	22	21

Table 1. Data for Example 1.

i	0	1	2	3	4	5	6	7	8	9	10	11	12
s_i	0	78	0	165	136	117	108	83	34	34	133	55	27
c_i	1	0	1	0	1	1	1	1	1	1	0	1	1
r_i	1	2	2	3	1	2	3	3	1	2	3	3	3

Table 2. Optimal solution with a collision for Example 1.

with three hoists for this example based on the above collision-free formulation. The optimal cycle time is 178.

Although the solution obtained satisfies (12) and (13), a collision between hoists 1 and 2 happens when hoist 1 is travelling from M_9 to M_4 and hoist 2 is travelling from M_{10} to M_1. To avoid possible collisions between any two hoists using a common zone, in addition to (12) and (13), the following two properties must hold. Note that $\sigma_k = \langle l_0^k l_1^k \cdots l_{N_k}^k \rangle$, $1 \le k \le G$, is the sequence of moves for hoist k corresponding to a cyclic schedule.

Property 1: In a collision-free schedule P, for any hoist k and for any two adjacent moves l_i^k and l_{i+1}^k such that $l_{i+1}^k < l_i^k$, for any hoist k' such that $k' < k$, if there exists a move $l_j^{k'}$ such that $l_{i+1}^k < l_j^{k'} < l_i^k$ and $s_{l_j^{k'}} < s_{l_{i+1}^k} < s_{l_{j+1}^{k'}}$, we must have $l_{j+1}^{k'} < l_{i+1}^k$ (figure 4a).

Property 2: In a collision-free schedule P, for any hoist k and for any two adjacent moves l_i^k and l_{i+1}^k such that $l_{i+1}^k > l_i^k$, for any hoist k' such that $k' > k$, if there exists a move $l_j^{k'}$ such that $l_i^k < l_j^{k'} < l_{i+1}^k$ and $s_{l_j^{k'}} < s_{l_{i+1}^k} < s_{l_{j+1}^{k'}}$, we must have $l_{j+1}^{k'} > l_{i+1}^k$ (figure 4b).

The correctness of Property 1 is straightforward. If $l_{j+1}^{k'} > l_{i+1}^k$, then hoist k will be before hoist k' at time $s_{l_{i+1}^k}$. Since $k' < k$, a collision between them must have happened. Similarly, for Property 2, if $l_{j+1}^{k'} < l_{i+1}^k$, hoist k will be after hoist k' at time $s_{l_{i+1}^k}$. Since $k' > k$, a collision between them must have happened.

It was found that only in about 2% of tested instances the solutions violate Properties 1 and 2 with the consideration of (12) and (13). This means they are very strict and necessary collision-free conditions. The next section will prove that Properties 1 and 2 can be relaxed in the branch-and-bound algorithm.

2.2.6. *Formulation of the tank capacity constraints*

The tank capacity constraints require that when a part arrives at a tank, the previous part should have been removed from that tank, since each tank can process at most one part at a time. If M_i contains no part at the beginning

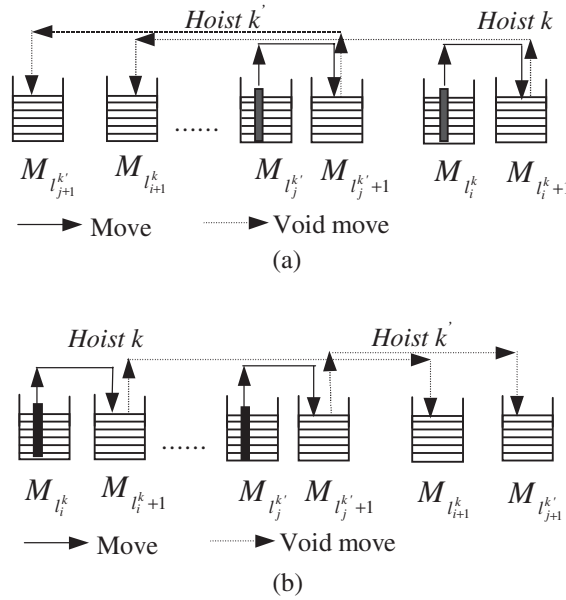


Figure 4. (a) Property 1 ($k' < k$); (b) Property 2 ($k' > k$).

of a cycle, i.e. $c_i = 0$, the tank capacity constraint is naturally satisfied. Otherwise, unloading a part from M_i must happen before loading another part into that tank.

If move $i-1$ and move i, $1 \le i \le N$, are performed by one hoist, in view of (6), the following condition must hold:

$$s_i + \theta_i + \delta_{i+1,i-1} \le s_{i-1}, \quad \text{for all } c_i = 1, (i, i-1) \in S.$$

If move $i-1$ and move i, $1 \le i \le N$, are performed by two hoists sharing the boundary tank M_i, i.e. $r_i > r_{i-1}$. According to (10), the following condition must hold:

$$s_i \le s_{i-1} + \theta_{i-1} - \varepsilon_i - \tau_i - (r_i - r_{i-1})\beta, \quad \text{for all } c_i = 1, (i, i-1) \in Q.$$

If move $i-1$ and move i, $1 \le i \le N$, are performed by two hoists sharing the segment of the track from M_{i-1} to M_{i+1}, i.e. $r_i < r_{i-1}$. In view of (12), the following condition must hold:

$$s_i + \theta_i + \delta_{i+1,i-1} + |r_{i-1} - r_i|\beta \le s_{i-1}, \quad \text{for all } c_i = 1, (i, i-1) \in D.$$

2.2.7. *Mathematical model*

To sum up, the single-track cyclic multi-hoist-scheduling problem can be formulated as:

$$\mathrm{P}_{\mathrm{chs}} : \min x$$

subject to

$$
\begin{aligned}
& a_i \le s_i + c_i x - s_{i-1} - \theta_{i-1} \le b_i, && \text{for all } i = 1, 2, \ldots, N \\
& s_i + \theta_i + \delta_{i+1,j} + |r_i - r_j|\beta \le s_j, && \text{for all } s_i < s_j,\ (i,j) \in S \cup D \qquad (17) \\
& s_j + \theta_j + \delta_{j+1,i} + |r_i - r_j|\beta \le x + s_i, && \text{for all } (i,j) \in S \cup D \qquad (18) \\
& s_i + \theta_i + \delta_{i+1,j} + |r_i - r_j|\beta \le x + s_j, && \text{for all } (i,j) \in S \cup D \qquad (19) \\
& s_i + \theta_i + \delta_{i+1,i-1} \le s_{i-1}, && \text{for all } c_i = 1, (i, i-1) \in S \\
& s_i \le s_{i-1} + \theta_{i-1} - \varepsilon_i - \tau_i - (r_i - r_{i-1})\beta, && \text{for all } c_i = 1, (i, i-1) \in Q \\
& s_i + \theta_i + \delta_{i+1,i-1} + |r_{i-1} - r_i|\beta \le s_{i-1}, && \text{for all } c_i = 1, (i, i-1) \in D,
\end{aligned}
$$

and the solution satisfies Properties 1 and 2. Note that (17) is a generalization of (6), (12) and (14); (18) and (19) are generalizations of (8), (9), (15) and (16).

3. Property analysis of the collision-free constraints

3.1. *Relaxation of Properties 1 and 2*

The previous section formulated the single-track cyclic multi-hoist-scheduling problem. The collision-free constraints are formulated as (12) and (13), as well as Properties 1 and 2. The properties are very difficult to deal with when designing an effective algorithm. They are more like two collision-checking rules. This section proves that due to some particular properties, Properties 1 and 2 can be relaxed in the proposed branch-and bound algorithm. If an optimal solution, due to relaxation of Properties 1 and 2, violates Properties 1 and 2, one can always obtain a collision-free optimal solution from the relaxed one.

Theorem 1: If a schedule $\mathrm{P} = \{\sigma_k | \sigma_k = \langle l_0^k l_1^k \cdots l_{N_k}^k \rangle, 1 \le k \le G\}$ violates Property 1, a collision-free schedule P' always dominates P with the same cycle time.

Proof: See appendix A.

Theorem 2: If a schedule $\mathrm{P} = \{\sigma_k | \sigma_k = \langle l_0^k l_1^k \cdots l_{N_k}^k \rangle, 1 \le k \le G\}$ violates Property 2, a collision-free schedule P' always dominates P with the same cycle time.

Proof: See appendix B.

Example 1 (continued): From table 2, $\sigma_1 = \langle 0, 8, 4 \rangle$, $\sigma_2 = \langle 2, 9, 1, 5 \rangle$ and $\sigma_3 = \langle 12, 11, 7, 6, 10, 3 \rangle$. By Property 1, a collision between hoists 1 and 2 happens during hoist 1 performing moves $(8, 4)$ and hoist 2 performing moves $(9, 1)$. According to Theorem 1, this collision can be solved by using hoist 1 instead of hoist 2 to perform move 1. Table 3 shows the collision-free optimal solution for this example. From table 3, $\sigma_1 = \langle 0, 8, 1, 4 \rangle$, $\sigma_2 = \langle 2, 9, 5 \rangle$ and $\sigma_3 = \langle 12, 11, 7, 6, 10, 3 \rangle$. The optimal cycle time remains 178. The optimal schedule is shown in figure 5.

i	0	1	2	3	4	5	6	7	8	9	10	11	12
s_i	0	78	0	165	136	117	108	83	34	34	133	55	27
c_i	1	0	1	0	1	1	1	1	1	1	0	1	1
r_i	1	1	2	3	1	2	3	3	1	2	3	3	3

Table 3. Collision-free optimal solution for Example 1.

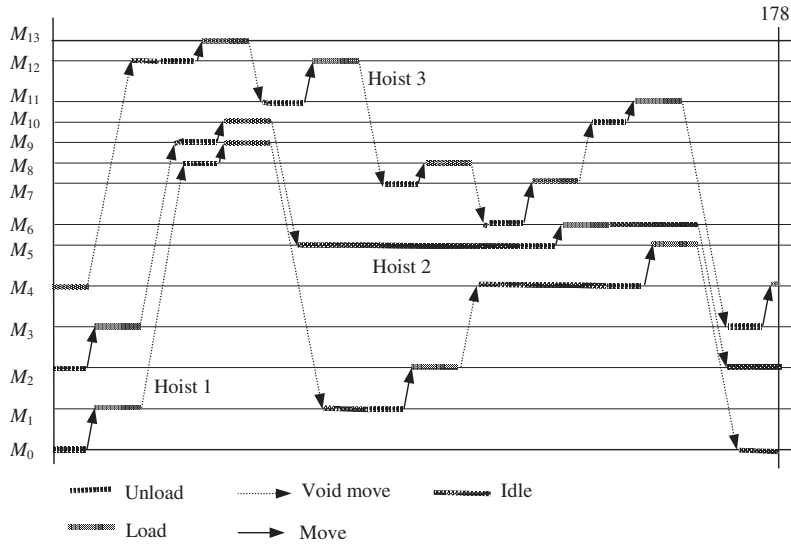


Figure 5. Optimal schedule for example 1.

In this example, hoists 1 and 2 share the zone from M_2 to M_9; hoists 2 and 3 use the overlapping zone from M_3 to M_{10}; hoists 1 and 3 share the zone from M_3 to M_9.

3.2. *Inconsistency problem in the formulation of the collision-free constraints*

For any two adjacent moves l_i^k and l_{i+1}^k performed by hoist k, due to the hoist capacity constraint:

$$s_{l_i^k} + \theta_{l_i^k} + \delta_{l_i^k+1, l_{i+1}^k} \leq s_{l_{i+1}^k}.$$

If $s_{l_i^k} + \theta_{l_i^k} + \delta_{l_i^k+1, l_{i+1}^k} < s_{l_{i+1}^k}$, upon completion of move l_i^k, hoist k has choices between leaving tank $l_i^k + 1$ as early as possible (at time $s_{l_i^k} + \theta_{l_i^k}$) and leaving tank $l_i^k + 1$ as late as possible (at time $s_{l_{i+1}^k} - \delta_{l_i^k+1, l_{i+1}^k}$). Therefore, in the formulation of the collision-free constraints, it is possible in some very particular cases that a hoist be required to leave a tank as early as possible to avoid a collision, and also be required to leave the tank as late as possible to avoid another collision. Thus, the inconsistency problem arises. The following shows that this inconsistency can be solved. Note that if $s_{l_i^k} + \theta_{l_i^k} + \delta_{l_i^k+1, l_{i+1}^k} = s_{l_{i+1}^k}$, no inconsistency problems will arise.

Consider four moves i, j, k, i', with $i < j < k < i'$, $r_j < r_i = r_{i'} < r_k$, and $s_i < s_j$, $s_k < s_{i'}$. Moves i and i' are two adjacent moves performed by hoist r_i (figure 6). Due to the hoist capacity constraint, the following relation holds:

$$s_i + \theta_{i+1} + \delta_{i+1, i'} < s_{i'}. \tag{20}$$

For hoists r_i and r_j using the common segment of the track from M_i to M_{j+1}, the following condition holds in order to avoid a collision:

$$s_i + \theta_i + \delta_{i+1, j} + (r_i - r_j)\beta \leq s_j. \tag{21}$$

To avoid a possible collision between hoists r_i and r_j, upon completion of move i, hoist r_i must leave M_{i+1} as early as possible if the following condition holds:

$$s_j < s_{i'} - \delta_{j, i'} + (r_i - r_j)\beta. \tag{22}$$

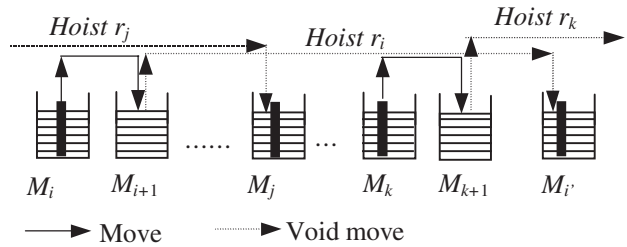


Figure 6. Inconsistency example with $r_j < r_i < r_k$.

This relation implies that hoist r_i cannot leave M_{i+1} as late as possible, since otherwise hoist r_i will arrive at M_j at time $s_{i'} - \delta_{j,i'}$, thus hoist r_i will arrive at M_j later than hoist r_j, which means a collision between hoists r_i and r_j must have happened. Similarly, for hoists r_k and r_i using the common segment of the track from M_k to $M_{i'+1}$, the following relation holds:

$$s_k + \theta_k + \delta_{k+1,i'} + (r_k - r_i)\beta \le s_{i'}. \tag{23}$$

To avoid a possible collision between hoists r_i and r_k, hoist r_i must arrive at $M_{i'}$ as late as possible if the following condition holds:

$$s_i + \theta_i + \delta_{i+1,k+1} < s_k + \theta_k + (r_k - r_i)\beta. \tag{24}$$

This relation implies that hoist r_i cannot leave M_{i+1} as early as possible, since otherwise hoist r_i will arrive at M_{k+1} at time $s_i + \theta_i + \delta_{i+1,k+1}$, thus hoist r_i will arrive at M_{k+1} earlier than hoist r_k, which means a collision between hoists r_i and r_k must have happened. For this example, hoist r_i is required to leave M_{i+1} as early as possible if (22) holds. At the same time, hoist r_i is required to arrive at $M_{i'}$ as late as possible if (24) holds. Therefore, the inconsistency problem arises. For this example, the inconsistency problem arises only if (20), (22) and (24) hold.

This inconsistency is solved as follows, provided that hoist r_i can pause in the track if necessary, which is very realistic since the hoists are computer controlled. Upon completion of a move (move i in this example), the hoist is always required to leave the tank (M_{i+1} in this example) as early as possible. If arrival at its next tank as early as possible would lead to a collision with another hoist, the hoist has to pause in the track to avoid a collision. To be more specific, for this example, hoist r_i will leave M_{i+1} at time $s_i + \theta_i$, and one has to consider two cases.

If $s_i + \theta_i + \delta_{i+1,k} \le s_k$, which means that when hoist r_i arrives at M_k, move k has not happened. As described above, in this case, hoist r_i is used instead of hoist r_k to perform move k, since in view of (23), upon completion of move k, hoist r_i has sufficient time to move to $M_{i'}$.

If $s_i + \theta_i + \delta_{i+1,k} > s_k$, which means that when hoist r_i arrives at M_k, move k has happened. By (24), hoist r_i has to pause somewhere before M_k to allow hoist r_k to finish move k. Otherwise, a collision will happen.

When hoist r_i moves from M_{i+1} to $M_{i'}$, it may also be waiting for other hoists after hoist r_i to finish a move, if necessary. Therefore, it is possible that hoist r_i needs to pause more than one time during the void move from M_{i+1} to $M_{i'}$. Note that it is also possible that more than one hoist waits for a hoist to finish a move.

4. Branch-and-bound algorithm

This section proposes a branch-and-bound algorithm based on the model presented in section 2. It consists of two nested branch-and-bound procedures: A and B. Procedure A implicitly enumerates all possible tank state distributions at the beginning of a cycle. Procedure B implicitly enumerates the sequences of hoist moves for each tank state distribution non-eliminated in procedure A.

4.1. *Branch-and-bound procedure A*

For each $n \leq N$, $Z_n = \{z_0, z_1, \ldots, z_n\}$ is called a partial tank state distribution. The branch-and-bound procedure A implicitly enumerates all possible tank state distributions in a cycle. Branching from a node corresponding to M_i, $0 \leq i \leq N-1$, consists of enumerating all possible tank states for M_{i+1}. Each tank can either contain a part or be empty at the beginning of a cycle, and the corresponding move from that tank to the next one can be performed by any one of G hoists. Therefore, each tank can have $2G$ different states. However, due to the single-track constraints, some feasibility-checking rules can be used to eliminate some infeasible nodes.

Theorem 3: Assume that no collisions happen during a cycle. Given a complete hoist distribution $R = \{r_0, r_1, \ldots, r_N\}$. For any j such that $0 \leq j \leq N$, (1) for any l such that $1 \leq l \leq r_j - 1$, there must be an i such that $0 \leq i < j$ and $r_i = l$; and (2) for any l such that $r_j + 1 \leq l \leq G$, there must be an i such that $j < i \leq N$ and $r_i = l$.

Proof: See appendix C.

Corollary 1: For any i such that $0 \leq i \leq G-1$, we have $1 \leq r_i \leq i+1$.

Corollary 2: For any i such that $N-G+1 \leq i \leq N$, we have $G-N+i \leq r_i \leq G$.

Corollary 3: Given a partial hoist distribution $R_n = \{r_0, r_1, \ldots, r_n\}$, where $0 \leq n \leq N$, for any j such that $0 \leq j \leq n$ and for any l such that $1 \leq l \leq r_j - 1$, there must be an i such that $0 \leq i < j$ and $r_i = l$.

A node, corresponding to a partial hoist distribution that violates any of the above corollaries can be eliminated from the enumeration tree. A node corresponding to a complete hoist distribution violating Theorem 3 can also be discarded. For example, a partial hoist distribution $R_4 = \{1, 1, 1, 3\}$ needs not to be considered in the enumeration tree. Such a partial hoist distribution cannot lead to a feasible tank state distribution.

The enumeration tree for the branch-and-bound procedure A is shown in figure 7. In enumeration tree A, a node with label $z_i = (0, r_i)$ means that M_i is empty at the beginning of a cycle and move i is performed by hoist r_i. Conversely, a node with label $z_i = (1, r_i)$ means that M_i has a part at the beginning of a cycle and move i is performed by hoist r_i.

A node on the nth, $0 \leq n \leq N$, level of the enumeration tree corresponds to a partial tank state distribution $Z_n = \{z_0, z_1, \ldots, z_n\}$, whose values are determined. For simplicity, define $S_n = \{(i,j) | r_i = r_j, i, j = 0, 1, \ldots, n\}$, $D_n = \{(i,j) | i < j, r_i > r_j$ or $i > j, r_i < r_j, i, j = 0, 1, \ldots, n\}$ and $Q_n = \{(i,j) | i < j, r_i < r_j$ or $i > j, r_i > r_j, i, j = 0, 1, \ldots, n\}$. By definition, $S_n \subseteq S$, $D_n \subseteq D$, $Q_n \subseteq Q$ for any $0 \leq n \leq N$. For a given Z_n, a lower bound of the cycle time is obtained by solving the following relaxed

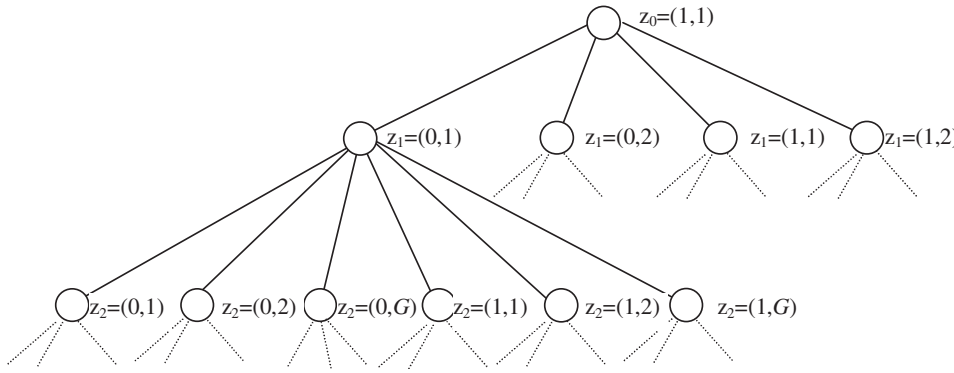


Figure 7. Enumeration tree A for the branch-and-bound algorithm.

linear programming problem:

$$P_{Z_n} : \min x$$

subject to

$$a_i \le s_i + c_i x - s_{i-1} - \theta_{i-1} \le b_i, \qquad \text{for all } i \le n \tag{25}$$

$$s_i + \theta_i + \delta_{i+1,i-1} \le s_{i-1}, \qquad \text{for all } c_i = 1, (i, i-1) \in S_n \tag{26}$$

$$s_i \le s_{i-1} + \theta_{i-1} - \varepsilon_i - \tau_i - (r_i - r_{i-1})\beta, \qquad \text{for all } c_i = 1, (i, i-1) \in Q_n \tag{27}$$

$$s_i + \theta_i + \delta_{i+1,i-1} + |r_{i-1} - r_i|\beta \le s_{i-1}, \qquad \text{for all } c_i = 1, (i, i-1) \in D_n \tag{28}$$

$$s_j + \theta_j + \delta_{j+1,i} + |r_i - r_j|\beta \le x + s_i, \qquad \text{for all } (i,j) \in S_n \cup D_n \tag{29}$$

$$s_i + \theta_i + \delta_{i+1,j} + |r_i - r_j|\beta \le x + s_j, \qquad \text{for all } (i,j) \in S_n \cup D_n. \tag{30}$$

The optimal value of the problem gives a lower bound of the cycle time. If the lower bound is greater than or equal to a known upper bound, the corresponding node can be eliminated. When enumeration tree A reaches a complete tank state distribution Z_N (i.e. Z), procedure B is activated.

4.2. *Branch-and-bound procedure B*

Procedure B seeks an optimal cyclic schedule for a given complete tank state distribution Z_N. Each enumeration tree B corresponds to a non-eliminated node on the Nth level in enumeration tree A. Procedure B implicitly enumerates the precedence relations between any two moves for the given complete tank state distribution.

In procedure B, each node corresponds to a set of partial precedence relations between two hoist moves, denoted by O, together with the constraints of problem P_{Z_N}. This node is denoted by (Z_N, O). The lower bound of the cycle time corresponding to node (Z_N, O) is obtained by solving the following linear programming problem:

$$P_{(Z_N, O)} : \min x$$

subject to

$$s_i + \theta_i + \delta_{i+1,j} + |r_i - r_j|\beta \le s_j, \quad \text{for all } (i,j) \in O, (i,j) \in S \cup D$$

and constraints (25–30) of problem P_{Z_N}.

At the root node of procedure B, $O = \phi$. The branching from a node (Z_N, O) is dependent on the solution of problem $P_{(Z_N, O)}$. If problem $P_{(Z_N, O)}$ has no solution or the lower bound is greater than or equal to a known upper bound, the corresponding node is eliminated. If the solution of problem $P_{(Z_N, O)}$, say $[x(Z_N, O), s(Z_N, O)]$, satisfies constraints (17), a feasible cyclic schedule with respect to the given complete tank state distribution Z_N has been obtained, and no further branching is needed from the corresponding node, since any feasible solution obtained by further branching from this node cannot be strictly better. Otherwise, there must be a pair of (i^*, j^*) such that the following relation holds:

$$s_{i^*}(Z_N, O) + \theta_{i^*} + \delta_{i^*+1, j^*} + |r_{i^*} - r_{j^*}|\beta > s_{j^*}(Z_N, O), s_{i^*}(Z_N, O) \leq s_{j^*}(Z_N, O),$$
$$\text{if}(i^*, j^*) \in S \cup D.$$

The branching consists of enumerating two precedence relations between these two moves i^* and j^*. That is, two partial precedence relation sets associated with the two subnodes are generated by adding to O either a precedence relation (i^*, j^*) or (j^*, i^*) (figure 8). A label such as (i_1, j_1) represents a precedence relation between moves i_1 and j_1, i.e. move i_1 precedes move j_1.

4.3. *Computation of lower bounds in procedures A and B using a graph-based algorithm*

The computation of lower bounds and detecting of infeasible solutions in branch-and-bound procedures A and B require the solution of specific linear programming problems (LPPs) P_{Z_n} and $P_{(Z_N, O)}$. In these LPPs, each linear inequality can be written equivalently in the form of $s_j - s_i \geq l_{i,j} - w_{i,j}x$, where $w_{i,j} = 0$, 1 or -1, $l_{i,j}$ is a real number. Due to this structure, each LPP can be transformed into a cycle time evaluation problem in a bivalued graph and can be solved using a graph-based polynomial algorithm (Chen *et al.* 1998). The computational complexities for solving problems P_{Z_n} and $P_{(Z_N, O)}$ using the graph-based algorithm are, respectively, $O(n^6)$ and $O(N^8)$.

5. Computational results

This section reports some computation results by comparing the proposed algorithm with the one proposed by Lei and Wang (1991), Lei *et al.* (1993) and Armstrong *et al.* (1996). The comparison was done by using a benchmark problem and randomly generated instances. In the present algorithm, hoists are allowed to use overlapping zones of the track while Lei and colleagues' algorithm divides a

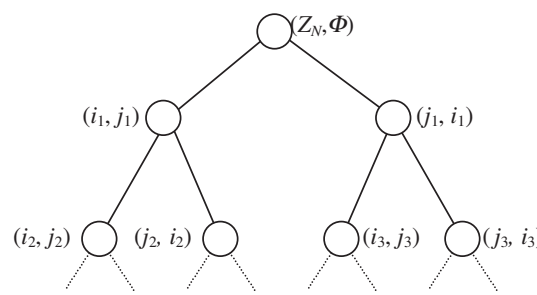


Figure 8. Enumeration tree B for the branch-and-bound algorithm.

Tank	1	2	3	4	5	6	7	8	9	10	11	12
a_i	150	90	120	90	30	60	60	45	130	120	90	30
b_i	200	120	180	125	40	120	120	75	∞	∞	120	60

Table 4. Minimum and maximum processing times for the P & U problem.

δ_{ij}	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0	11	14	16	14	19	22	24	26	29	6	8	10
1	11	0	2	5	2	8	10	13	15	17	10	3	1
2	14	2	0	2	0	5	8	10	13	15	12	6	3
3	16	5	2	0	2	3	5	8	10	13	15	8	6
4	14	2	0	2	0	5	8	10	13	15	12	6	3
5	19	8	5	3	5	0	3	5	7	10	18	11	9
6	22	10	8	5	8	3	0	2	5	7	20	14	11
7	24	13	10	8	10	5	2	0	2	5	23	16	14
8	26	15	13	10	13	7	5	2	0	2	25	19	16
9	29	17	15	13	15	10	7	5	2	0	27	21	19
10	6	10	12	15	12	18	20	23	25	27	0	7	9
11	8	3	6	8	6	11	14	16	19	21	7	0	2
12	10	1	3	6	3	9	11	14	16	19	9	2	0
θ_i	31	22	22	22	25	23	22	22	22	47	27	22	30

Table 5. Move times for the P & U problem.

line into non-overlapping zones, and each hoist is assigned to a zone. In the remainder, the present algorithm and Lei and colleagues' are, respectively, called the 'overlapping partition' algorithm and the 'non-overlapping partition' algorithm.

The P & U problem (Phillips and Unger 1976) has become a well-recognized benchmark in almost all the literature addressing cyclic single-hoist scheduling. As mentioned above, the present paper studies a unidirectional production line. It is assumed that the part flow in the P & U problem is unidirectional. For this problem, $\varepsilon_i = 8.5$, $\tau_i = 11.5$, for any $0 \leq i \leq 12$. Tables 4 and 5 give the processing times and move times for the P & U problem, respectively.

Table 6 gives the optimal multi-hoist cycle times for the P & U problem by the 'overlapping partition' algorithm and by the 'non-overlapping partition' algorithm. If a constraint $r_{i+1} \geq r_i$, for all $0 \leq i \leq N-1$, is added to the proposed mathematical model, the present algorithm is reduced to one in which the use of a common zone of the track by multiple hoists is not allowed, i.e. the non-overlapping partition algorithm. The data in the row 'Non-overlapping partition' in table 6 were obtained by that way, which were also reported by Armstrong *et al.* (1996). Table 6 shows that improvements in the optimal cycle times by multi-hoist schedules are very significant. It can also be seen that the optimal cycle times by using the overlapping partition algorithm are less than those by using the non-overlapping partition algorithm.

In addition, three sets of randomly generated instances were used to evaluate the performance of the algorithm. The instances were generated using integer uniform distributions. In the following, let $U(c_1, c_2)$ be a positive integer drawn randomly from a uniform distribution with lower and upper bounds c_1 and c_2. The first set of test instances was generated using the following parameters: $a_i = U(50, 150)$,

	$G = 1$	$G = 2$	$G = 3$	$G = 4$
Non-overlapping partition	521	251	217	215
Overlapping partition	521	251	198.5	170.5

Table 6. Optimal multi-hoist cycle times for the P & U problem.

	Non-overlapping			Overlapping			O/N		
G	2	3	4	2	3	4	2	3	4
$N = 8$	2.69	3.21	3.47	2.77	3.44	3.61	1.03	1.07	1.04
$N = 10$	2.93	4.16	4.51	3.18	4.52	4.95	1.08	1.09	1.10
$N = 12$	3.58	5.73	6.59	3.76	6.16	7.23	1.05	1.07	1.10
$N = 14$	4.26	7.35	9.12	4.52	8.22	10.08	1.06	1.12	1.06
$N = 16$	3.69	6.68	9.54	4.02	7.27	10.32	1.09	1.09	1.08

Table 7. Computational results for the first set of instances.

Q:	2			6			10		
G:	2	3	4	2	3	4	2	3	4
Non-overlapping	3.40	5.59	6.36	2.73	5.18	7.41	2.08	4.29	6.32
Overlapping	3.70	6.06	6.97	2.97	5.61	7.85	2.15	4.43	6.56
O/N	1.09	1.09	1.10	1.09	1.08	1.06	1.03	1.03	1.04

Table 8. Computational results for the second set of instances.

$b_i = a_i + U(10, 50)$, $\delta_{i,i+1} = U(2, 7)$, $\delta_{i,j} = \delta_{j,i} = \sum_{k=i}^{j-1} \delta_{k,k+1}$, $\varepsilon_i = 10$, $\tau_i = 10$, $\theta_i = \delta_{i,i+1} + 20$, $0 \le i \le N$, $N \in \{8, 10, 12, 14, 16\}$. The second set of test instances was generated by using the following parameters: $N = 12$, $a_i = U(50, 150)$, $b_i = a_i + U(10, 50)$, $\delta_{i,i+1} = U(Q, Q + 5)$, $\delta_{i,j} = \delta_{j,i} = \sum_{k=i}^{j-1} \delta_{k,k+1}$, $\varepsilon_i = 10$, $\tau_i = 10$, $\theta_i = \delta_{i,i+1} + 20$, $0 \le i \le N$, $Q \in \{2, 6, 10\}$. Parameter Q controls the relative importance of the processing times with respect to the hoist move times. The larger Q is, the greater chance the hoists become bottlenecks. Parameters used to generate the third set of problems were $N = 12$, $a_i = U(50, 150)$, $b_i = a_i + U(Y, 40 + Y)$, $\delta_{i,i+1} = U(2, 7)$, $\delta_{i,j} = \delta_{j,i} = \sum_{k=i}^{j-1} \delta_{k,k+1}$, $\varepsilon_i = 10$, $\tau_i = 10$, $\theta_i = \delta_{i,i+1} + 20$, $0 \le i \le N$, $Y \in \{10, 60, 110\}$. Parameter Y controls the width of the time windows. The larger Y is, the wider the time windows. In the instance generation, four values of G were considered: 1, 2, 3, 4. For each combination of parameters, 20 instances were generated. (The detailed test data can be obtained from the authors upon request.)

Tables 7–9, respectively, show the computational results for the first, second and third sets of test instances. The rows and/or columns 'Non-overlapping' (respectively 'Overlapping') provide with the average improvements in the optimal cycle times achieved by the multi-hoist schedules with respect to the single-hoist schedules, i.e. Optimal single-hoist cycle time/Optimal multi-hoist cycle time, by using the non-overlapping partition algorithm (respectively overlapping partition algorithm). The rows and/or columns 'O/N' give the ratios between these two rows and/or columns.

Y:	10			60			110		
G:	2	3	4	2	3	4	2	3	4
Non-overlapping	3.40	5.59	6.36	2.71	3.52	3.76	2.35	2.88	3.05
Overlapping	3.70	6.06	6.97	2.76	3.75	4.09	2.36	3.04	3.31
O/N	1.09	1.09	1.10	1.02	1.07	1.09	1.01	1.06	1.09

Table 9. Computational results for the third set of instances.

	N	8	10	12	14	16
$G = 1$	B & B nodes	161.1	228	312.8	391.9	395.1
	pruned nodes	88.05	126.4	173.3	218.4	222.75
	P/B (%)	54.66	55.44	55.40	55.73	56.38
2	B & B nodes	1100.9	5874.4	23779.3	83712.1	198823.5
	pruned nodes	736.8	3772.35	14737.35	49643.95	118397.7
	P/B (%)	66.93	64.22	61.98	59.30	59.55
3	B & B nodes	1822.5	9738.0	59714.6	402946.1	1579386.5
	pruned nodes	1197.05	6743.35	40499.15	255150.5	984900.15
	P/B (%)	65.68	69.25	67.82	63.32	62.36
4	B & B nodes	2428.5	17138.4	80448.1	447384.7	3027274.2
	pruned nodes	1283.7	10752.1	56063.4	302633.9	2019343.35
	P/B (%)	52.86	62.74	69.69	67.65	66.71

Table 10. Statistics on the B & B size for the first set of instances.

These tables show that the improvements in the optimal cycle times achieved by the multi-hoist schedules are very significant. For example, for the two-hoist schedules, the improvements range from 2.08 to 4.52. Table 7 shows that the improvements achieved by the multi-hoist schedules increase in general with N. The larger the problem size, the more improvements. It can also be seen in tables 8 and 9 that the improvements decrease with Q and Y. The greater the chance that the hoists become bottlenecks or the wider the time windows, the less the improvements. The improvements in the optimal cycle times by using the 'overlapping partition' algorithm are greater than those by using the 'non-overlapping partition' algorithm. As mentioned above, the lot sizes of parts in electroplating lines are usually very large, therefore, even a small reduction in the cycle time may lead to a significant improvement in the production volume.

Table 10 gives the statistics on the B & B size for the first set of test instances. The columns 'B & B nodes' and 'Pruned nodes', respectively, represent the average total nodes and average pruned nodes of the branch-and-bound trees. The column (P/B) gives the ratio between them. Nearly 54–69% nodes are pruned nodes. This means that the lower bound and the branching scheme developed here are very efficient.

Table 11 shows the average computation times (CPU s) for the first set of test instances on a Pentium III 450-MHz processor. It seems that the average computation times depend rather on the problem size than on the number of hoists.

N	8	10	12	14	16
$G = 1$	0.02	0.04	0.05	0.07	0.08
2	0.14	0.77	3.54	14.86	39.90
3	0.16	1.08	7.60	60.69	265.10
4	0.14	1.43	8.98	60.90	471.34

Table 11. Average computation times for the first set of instances.

6. Conclusion

This paper proposes an analytical model for the single-track multi-hoist-scheduling problem in a PCB electroplating facility. The collision-free constraints are formulated as disjunctive inequalities and two collision-checking properties. It is proved that a schedule violating these two properties due to relaxation of them is always dominated by a collision-free one. Therefore, these two properties are relaxed in the branch-and-bound algorithm. Computational results show that the improvements in the optimal cycle times achieved by the multi-hoist schedules by using our algorithm are greater than those by using the algorithm proposed by Lei and Wang (1991), Lei *et al.* (1993) and Armstrong *et al.* (1996). The lower bound and the branching scheme developed in the present paper are very efficient.

We investigated to extend the proposed approach to the more complex multi-directional production line. In the latter case, in addition to the collisions considered in this paper, it has to consider the possible collisions between any two hoists during their performances of two loaded moves.

Acknowledgements

This work was supported by a postdoctoral grant from the French Ministry of National Education. Part of the work was done in 1999 when the first author was supported by the French–Israeli cooperation grant 'Factory of the Future'. The authors are grateful to anonymous reviewers for their helpful remarks.

Appendix A. Proof of Theorem 1

Due to the assumption that the schedule $\mathrm{P} = \{\sigma_k | \sigma_k = \langle l_0^k l_1^k \cdots l_{N_k}^k \rangle, 1 \le k \le G\}$ violates Property 1, there must exist hoists k and k', and corresponding moves (l_i^k, l_{i+1}^k) and $(l_j^{k'}, l_{j+1}^{k'})$ such that $k' < k$, $l_{i+1}^k < l_j^{k'} < l_i^k$, $s_{l_j^{k'}} < s_{l_{i+1}^k} < s_{l_{j+1}^{k'}}$ and $l_{j+1}^{k'} > l_{i+1}^k$. In this case, a collision between hoists k and k' happens when hoist k moves from tank $l_i^k + 1$ to tank l_{i+1}^k.

Since $s_{l_j^{k'}} < s_{l_{i+1}^k}$, $l_{i+1}^k < l_j^{k'}$, $k' < k$, $(l_{i+1}^k, l_j^{k'}) \in D$, one must have:

$$s_{l_j^{k'}} + \theta_{l_j^{k'}} + \delta_{l_j^{k'}+1, l_{i+1}^k} + (k - k')\beta \le s_{l_{i+1}^k}.$$

Due to $(k - k')\beta > 0$, it follows from this relation that:

$$s_{l_j^{k'}} + \theta_{l_j^{k'}} + \delta_{l_j^{k'}+1, l_{i+1}^k} < s_{l_{i+1}^k}. \tag{A1}$$

Since $s_{l_{j+1}^{k'}} > s_{l_{i+1}^k}$, $l_{j+1}^{k'} > l_{i+1}^k$, $k' < k$, $(l_{i+1}^k, l_{j+1}^{k'}) \in D$, one also must have:

$$s_{l_{i+1}^k} + \theta_{l_{i+1}^k} + \delta_{l_{i+1}^k+1, l_{j+1}^{k'}} + (k - k')\beta \le s_{l_{j+1}^{k'}}.$$

It follows from this relation that:

$$s_{l_{i+1}^k} + \theta_{l_{i+1}^k} + \delta_{l_{i+1}^k+1, l_{j+1}^{k'}} < s_{l_{j+1}^{k'}}. \tag{A2}$$

From (A1) and (A2), hoist k' is idle while hoist k is performing move l_{i+1}^k. Relation (A1) says that if move l_{i+1}^k is performed by hoist k' instead of hoist k, hoist k' has sufficient time to move to tank l_{i+1}^k upon completion of move $l_j^{k'}$. Relation (A2) says that hoist k' has sufficient time to move to tank $l_{j+1}^{k'}$ to perform its next scheduled move upon completion of move l_{i+1}^k. As a result, move l_{i+1}^k can be performed by hoist k' instead of hoist k. Therefore, the collision between hoist k' and hoist k disappears by using hoist k' instead of hoist k to perform move l_{i+1}^k.

If there exists more than one pair of (l_i^k, l_{i+1}^k) and $(l_j^{k'}, l_{j+1}^{k'})$ such that $k' < k$, $l_{i+1}^k < l_j^{k'} < l_i^k$, $s_{l_j^{k'}} < s_{l_{i+1}^k} < s_{l_{j+1}^{k'}}$ and $l_{j+1}^{k'} > l_{i+1}^k$, for any pair of (l_i^k, l_{i+1}^k) and $(l_j^{k'}, l_{j+1}^{k'})$, the collision between hoists k' and k can always be solved by using hoist k' instead of hoist k to perform move l_{i+1}^k. It should be noted that this replacement of hoists, which lead to pausing of hoist k on tank $l_i^k + 1$ for a long period, may cause new collisions between hoist k and hoist k'' such that $k'' > k$. However, the new collisions can also be solved by using hoist k instead of hoist k'' to perform the corresponding move. This process will stop when either a new collision due to the replacement of the hoists does not happen or k'' becomes G. Thus, we have Theorem 1.

Appendix B. Proof of Theorem 2

Proof: The proof of Theorem 2 is similar to the one for Theorem 1. For any hoists k and k', any pair (l_i^k, l_{i+1}^k) and any pair $(l_j^{k'}, l_{j+1}^{k'})$ such that $k' > k$, $l_i^k < l_j^{k'} < l_{i+1}^k$, $s_{l_j^{k'}} < s_{l_{i+1}^k} < s_{l_{j+1}^{k'}}$, and $l_{j+1}^{k'} < l_{i+1}^k$, the collision between hoists k' and k can always be solved by using hoist k' instead of hoist k to perform move l_{i+1}^k. If this replacement causes new collisions between hoist k and hoist k'' such that $k'' < k$. The new collisions can also be solved by using hoist k instead of hoist k'' to perform the corresponding move. This process will stop when either a new collision due to the replacement of the hoists does not happen or k'' becomes 1. Thus, we have Theorem 2.

Appendix C. Proof of Theorem 3

Proof: As described in section 2, all the hoists are numbered in the same order as the tanks in a production line. Furthermore, at any time of a cycle, any hoist is either performing a loaded move, or performing a void move from one tank (to which it has just loaded a part) to another (from which it will unload a part), or waiting on a tank for completion of a part. Thus, with the assumption that no collisions happen during a cycle, given a complete hoist distribution, for each j such that $0 \le j \le N$, hoists $1, 2, \ldots, r_j - 1$ must be before hoist r_j, and hoists $r_j + 1$, $r_j + 2, \ldots, G$ must be after hoist r_j. Thus, we have theorem 3.

References

Armstrong, R., Gu, S. and Lei, L., 1996, A greedy algorithm to determine the number of transporters in a cyclic electroplating process. *IIE Transactions*, **28**, 347–355.

Bloch, C. and Manier, M. A., 1999, Notation and typology for the hoist scheduling problem. Proceedings of the IEEE SMC Conference, Tokyo, Japan, pp. 475–480.

Che, A., Chu, C. and Chu, F., 2002, Multi-cyclic hoist scheduling with constant processing times. *IEEE Transactions on Robotics and Automation*, **18**, 69–80.

Che, A., Chu, C. and Levner, E., 2003, A polynomial algorithm for 2-degree cyclic robot scheduling. *European Journal of Operational Research*, **145**, 31–44.

Chen, H., Chu, C. and Proth, J.-M., 1998, Cyclic scheduling of a hoist with time window constraints. *IEEE Transactions on Robotics and Automation*, **14**, 144–152.

Co, C. G. and Tanchoco, J. M. A., 1991, A review of research on AGVS vehicle management. *Engineering Costs and Production Economics*, **21**, 35–42.

Karzanov, A. V. and Livshits, E. M., 1978, Minimal quantity of operators for serving a homogeneous linear technological process. *Automation and Remote Control*, **3**, 162–169.

Kats, V. and Levner, E., 1997, Minimizing the number of robots to meet a given cyclic schedule. *Annals of Operations Research*, **69**, 209–226.

Kats, V., Levner, E. and Meyzin, L., 1999, Multiple-part cyclic hoist scheduling using a sieve method. *IEEE Transactions on Robotics and Automation*, **15**, 704–713.

Lamothe, J., 1996, Une approche pour l'ordonnancement dynamique d'un atelier de traitement des surfaces. PhD thesis, l'ENSAE, France.

Lamothe, J., Thierry, C. and Delmas, J., 1996, *A Multihoist Model for the Real Time Hoist Scheduling Problem* (Lille: IMACS, CESA), pp. 461–466.

Lee, C., Lei, L. and Pinedo, M., 1997, Current trends in deterministic scheduling. *Annals of Operations Research*, **70**, 1–41.

Lei, L., Armstrong, R. and Gu, S., 1993, Minimizing the fleet size with dependent time window and single-track constraints. *Operation Research Letters*, **14**, 91–98.

Lei, L. and Wang, T. J., 1989, *A Proof: The Cyclic Hoist Scheduling Problem is NP-Hard*. Working Paper #89-0016, Rutgers University.

Lei, L. and Wang, T. J., 1991, The minimum common-cycle algorithm for cyclic scheduling of two material handling hoists with time window constraints. *Management Science*, **37**, 1629–1638.

Lei, L. and Wang, T. J., 1994, Determining optimal cyclic hoist schedules in a single-hoist electroplating line. *IIE Transactions*, **26**, 25–33.

Phillips, L. W. and Unger, P. S., 1976, Mathematical programming solution of a hoist scheduling program. *IIE Transactions*, **8**, 219–225.

Shapiro, G. W. and Nuttle, H. W., 1988, Hoist scheduling for a part electroplating facility. *IIE Transactions*, **20**, 157–167.

Song, W., Zabinsky, Z. B. and Storch, R. L., 1993, An algorithm for scheduling a chemical processing tank line. *Production Planning and Control*, **4**, 323–332.

Sun, T., Lai, K., Lam, K. and So, K., 1994, A study of heuristics for bidirectional multi-hoist production scheduling systems. *International Journal of Production Economics*, **33**, 207–214.

Varnier, C., Bachelu and Baptiste, P., 1997, Resolution of the cyclic multi-hoists scheduling problem with overlapping partitions. *INFOR (Information Systems and Operations Research)*, **35**, 277–284.