Optimal Multi-Degree Cyclic Scheduling of Re-entrant Electroplating Lines Including Multiple Hoists without Overlapping

Xin Li[1], Felix T. S. Chan[2*], and Ben Niu[3]
Department of Industrial and Systems Engineering,
The Hong Kong Polytechnic University, Hong Kong[1,2,3]
College of Management, Shenzhen University, China[3]
shenelee@gmail.com[1], f.chan@polyu.edu.hk[2], drniuben@gmail.com[3]
**Corresponding Author*

Received 26 March 2015; received in revised form 12 June 2015; accepted 30 June 2015

Abstract
Automated electroplating lines are typical integrated manufacturing systems consisting of processing machines and material handling devices, i.e. tanks and hoists respectively. Printed circuit boards, usually known as parts, are produced in a number of tanks following the given process flow. Reentrance will occur when a part is performed in the same tanks more than once due to the process requirement. Since parts are transported between tanks by hoists, the hoists usually create bottlenecks. Multiple hoists, instead of a single hoist, are largely used to balance the line and improve the throughput. Since hoists move along the same overhead track, they cannot cross over each other. This paper considers multi-hoist scheduling problems during the reentrance of automated electroplating lines. The objective is to maximize the throughput of the line, the equivalent to minimizing the cycle time, by obtaining optimal schedules. In order to tackle the collisions between the multiple hoists, the principle of non-overlapping is applied. Since identical parts are commonly arranged to be produced at the same time in a line during a period, cyclic scheduling is applied to deal with the scheduling problems due to easy implementation. This paper considers multi-degree cycles instead of a simple cycle, i.e. 1-degree cycles. In summary, this paper deals with the scheduling problems in a much more complicated scenario, where reentrance, multiple hoists and multi-degree cycles are included. To our knowledge, these complications have not been dealt with before. To obtain optimal schedules, the operations in the lines are analyzed in detail. Based on this, a mixed integer linear programming model is formulated to solve the scheduling problems in this scenario. This is the main contribution of this work. Finally, a numerical example, solved using the commercial software ILOG CPLEX, is applied to illustrate the model proposed and to show the benefits of multi-degree cycles compared with simple cycles.

Keywords: Multi-hoist scheduling problem, multi-degree cycles, reentrant electroplating lines, mixed integer linear programming, processing time window

1. Introduction

A typical integrated manufacturing system is the automated electroplating line for printed circuit board (PCBs) production. PCBs are knows as parts. A number of stages (i.e. stage 1, stage 2, …, stage n) are required to complete the parts in the lines, and the process stages are performed in related tanks, which are arranged in a line from left to right. A special condition in automated electroplating lines is where there are no buffers between the tanks. When a part completes its processing in a tank for a specific stage, the part should be picked up and transported to a tank for the next stage. Parts are transported between the tanks by hoists. Therefore, a part is either being produced in a tank or held by a hoist for transportation.

Moreover, each tank can produce, at most, one part at a time, known as tank capacity constraints. Similarly, each hoist can only hold, at most, one part at a time. Hence, a hoist should be available when the hoist is scheduled to transport a part, known as hoist availability constraints. Since all parts are transported by a shared hoist(s), the hoists usually become the bottleneck resource. Therefore, more than one hoist is generally applied to the lines so as to balance the production and improve the throughput. In multi-hoist lines, these multiple hoists share and move on the same overhead track. Hoists cannot cross over each other. Therefore, more constraints are required to avoid any conflicts between the hoists. In addition, the practical processing times of the parts are within given time ranges, known as time window constraints. There are other scenarios relevant to the processing times. When the upper bound is infinite, time window constraints are not required. When, the upper bound is exactly equal to the lower bound, it is known as a no-wait scenario. These two scenarios can be viewed as special cases of scenarios with time window constraints.

Moreover, there are other more complicated lines. For instance, some tanks may need to be visited more than once in accordance with the process flow, i.e. the same tanks perform more than one stage. This is also known as reentrance. This paper considers multi-hoist lines with reentrance.

In practice, the production flow usually arranges identical parts to be produced during a specific period. Cyclic scheduling is commonly used due to its easy implementation. The time duration of a cycle is called the cycle time. The degree is used to describe the number of parts that are inserted and finished during a cycle. 1-degree cycles are also known as simple cycles. One part is inserted and is completed during a simple cycle. A K-degree cycle (i.e. multi-degree cycle) produces K parts. In order to compare a K-degree cycle with a simple cycle, the mean cycle time is defined and calculated based on dividing the cycle time of the K-degree cycle by K. The mean cycle time is the (mean/average) time required to produce a part. For a given degree, the objective is to obtain scheduling that minimizes the cycle time, i.e. maximizes the throughput.

Since hoists are applied in automated electroplating lines, scheduling problems in these lines are known as the hoist scheduling problem (Philips and Unger, 1976; Manier and Bloch, 2003; Lopez and Roubellat, 2008).

This paper deals with the hoist scheduling problem by considering multiple hoists and reentrance together. Time window constraints are also considered. In addition, multi-degree cycles are analyzed. To our knowledge, this is the first work focusing on this special scenario (i.e. multi-hoist, reentrance, and multi-degree cycles). The objective is to obtain schedules that will minimize the (mean) cycle time. To achieve this, the problem is modelled based on the mixed integer linear programming approach. Detailed operations of the lines are also analyzed. To deal with this complicated problem, a revised scenario without reentrance was analyzed first. This is the necessary condition of the complicated scenario. Then, the reentrance was analyzed. The problem is modelled by an MILP formulation. An instance of the model is solved using ILOG CPLEX.

The remainder of this paper is organized as follows. Section 2 reviews related work on the hoist scheduling problem. Section 3 describes the special scheduling problem dealt with in this paper. Corresponding notations are listed. Section 4 compares the scheduling problem considered in this paper with the revised scenario, where reentrance is relaxed. The revised scenario has been dealt with by Li and Fung (2013a). For completeness, constraints of the revised scenario are formulated based on Li and Fung's (2013a) work. Section 5 analyzes operations concerning reentrance and develops corresponding constraints. In section 6, an example is used to illustrate the model proposed. Finally, this paper is concluded in section 7.

2. Literature Review

Philips and Unger (1976) proposed the first mathematical programming model (i.e. a mixed integer linear programming model) for the hoist scheduling problem. Shapiro and Nuttle (1988) developed a branch and bound approach to deal with the same problem. Since then, mathematical programming approaches, e.g. mixed integer linear programming (MILP) and branch and bound (B&B), have been increasingly applied to solve complicated scheduling problems in diverse scenarios.

Liu et al. (2002) proposed a comprehensive mixed integer programming model for the complicated lines with reentrance and parallel tanks in a simple cycle. Steiner and Xue (2005) reviewed scheduling problems in reentrant robotic cells, which have the same configuration as re-entrant electroplating lines. As for multi-hoist lines, Leung et al. (2004) developed the first mixed integer linear programming model in simple cycles. Middle-size instances (e.g. less than 20 tanks) can be solved using Leung et al.'s model with reasonable computational times. For large-size instances (e.g. 24 tanks), Zhou and Li (2009) proposed a new mixed integer linear programming model based on the no overlapping rule. In this way, the hoist assignment is determined first and final schedules are obtained using the MILP model. In fact, schedules with the no overlapping rule may not achieve global optimal solutions, but computational times are saved, i.e. large-size instances can be solved in reasonable time. Che et al. (2014) developed the MILP model to improve Leung et al.'s model. Jiang and Liu (2014) proposed a new MILP model and a B&B approach so as to deal with multi-hoist lines in simple cycles.

As for multi-degree cycles, Zhou et al. (2012) formulated the first MILP model for basic lines, i.e. the lines considered by Phillips and Unger (1976). Li and Fung (2014) developed a new MILP model for basic lines in multi-degree cycles. Then, Li and Fung (2014) extended their model to lines with reentrance. There is only one single hoist. For multi-hoist lines, Li and Fung (2013a) formulated the MILP model based on the no overlapping rule in multi-degree cycles. General multi-hoist lines were dealt with in Li and Fung (2013b) using the MILP model. In Li and Fung (2013a, 2013b), reentrance (or parallel tanks), were not considered with the multiple hoists at the same time. MILP models were also applied to deal with scenarios with multiple part types (Lei et al. 2014; El Amraoui et al. 2013b).

Heuristics and evolution algorithms have also been used to deal with the hoist scheduling problem. Zhou and Li (2008) proposed a heuristic algorithm to deal with two-hoist lines in simple cycles. El Amraoui et al. (2013a) developed a genetic algorithm for single-hoist scheduling problems with time window constraints. In addition, no-wait scenarios were solved in polynomial time (Che et al. 2002, Che et al. 2009, and Che et al. 2012).

Consequently, this paper deals with the multi-degree hoist scheduling problem and considers multiple hoists and reentrance together using mathematical programming (i.e. mixed integer linear programming). It extends Li and Fung's work (2013a, 2013b) which only deals with multi-degree hoist scheduling with multiple hoists without the occurrence of reentrance. Moreover, it is more complicated when compared with EI Amraoui's (2013a, 2013b) work that only deals with single hoist scheduling problems.

3. Problem Description and Notation

This paper deals with the multi-hoist line with reentrance, as shown in Figure 1. There are n stages indexed as $S_1, S_2, ..., S_n$ performed in n - 1 tanks numbered as 1, 2, ..., n - 1 from left to right. The indexes of the tanks increase from left to right. Except for the reentrant stage, a later stage is performed in a tank with a larger index, i.e. a tank that is further to the left. Moreover, stages p-1 and p+1 are both performed in tank p-1 and stage p+1 is the reentrant stage. *H* hoists transport parts between tanks to complete the corresponding stages. S_{p-1} and S_{p+1} use the same tank *p* - 1. To avoid conflict between the hoists, the no overlapping rule is applied in this paper, i.e. hoist h picks up parts processed in stage $l_{h-1}+1$ to stage l_h. Moreover, it is assumed that tank p-1 and tank p are aligned with the same hoist.

In order to express the scheduling problem exactly and formulate the MILP model, the parameters and decision variables are listed as follows. These notations are adopted from previous work (Phillips & Unger, 1976; Liu et al., 2002; Leung et al., 2004; Zhou & Li, 2009; Li & Fung 2013a, 2013b, 2014).

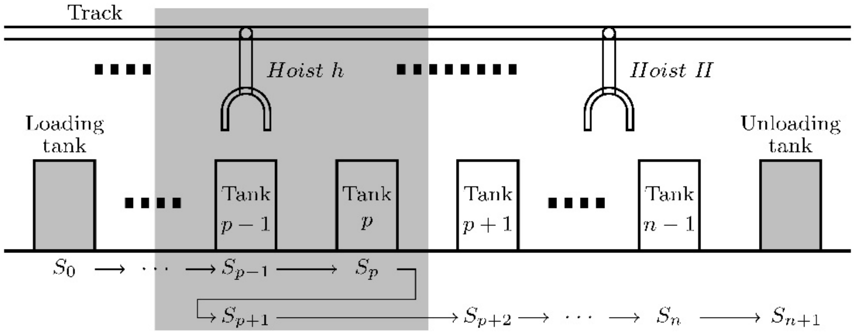

Figure 1: A Multi-hoist Line with Reentrance

3.1 Given Parameters

n: the number of tanks. The tanks are labeled 1; 2; …; *n*; the loading tank is numbered as 0 and the unloading tank is numbered as *n* + 1;

p: the stage p - 1 and stage p + 1 use the same tank *p* - 1;

H: the number of hoists;

K: the number of parts entering and exiting the line within a cycle, i.e. the degree of a cycle;

move (k,i): the hoist move when transporting a part from stage *i* to stage *i*+1 for the *k*-th time;

l_h: the parameter indicating the hoist assignment. Hoist 1 is responsible for moves from 0 to l_1 (to move out of tank 0 to l_1) and hoist h for moves $l_{h-1}+1$ to l_h;

L_i: the minimum amount of processing time a part requires in stage *i*;

U_i: the maximum amount of processing time a part is permitted in stage *i*;

a_i: the time required to unload a part from stage *i* corresponding to a *move(*, i)*;

b_i: the time required to unload a part from stage *i* + 1 corresponding to a *move(*, i)*;

d_i: the travel time for the hoist carrying a part from stage *i* to stage *i*+1 including the unloading time (a_i) and the loading time (b_i);

$e_{i,j}$: the travel time from stage *i* to stage *j* for an empty hoist;

Δ: a very small positive number;

M: a very large positive number

3.2 Decision Variables

T: cycle time;

$t_{k,i}$: the starting time of *move(k; i);*

$t_{min\,1}$: the starting time of the last *move(K, i)* within a cycle for hoist 1;

$t_{min\,h}$: the starting time of the first move(1, i) within a cycle for hoist h, h = 2, …, H;

$t_{\max h}$: the starting time of the last move(K, i) within a cycle for hoist h, h = 2, …, H;

x_i: is equal to 1 if the move(K, i) is the last move for hoist 1; otherwise, it is equal to 0, where i = 1, 2, …, l_1;

z_i^h : is equal to 1 if the move(K; i) is the last move for hoist h; otherwise, it is equal to 0, where h = 2, 3, …, H and $i = l_{h-1}, l_{h-1} + 1, \dots, l_h$;

s_j^h : is equal to 1 if the move(1, j) is the last move for hoist h; otherwise, it is equal to 0, where h = 2, 3, …, H and $j = l_{h-1}, l_{h-1} + 1, \dots, l_h$;

$w_{i,j}^h$: is equal to 1 if the move(1, j) is the last move and the move(K i) is the last move for hoist h; otherwise, it is equal to 0, where h = 2, 3, …, H and $i, j = l_{h-1}, l_{h-1} + 1, \dots, l_h$;

$y_{r,i;u,j}$: is equal to 1 if $t_{r,i} \le t_{u,j}$; otherwise, it is equal to 0, where $h = 1, 2, \dots, H$, $i, j = l_{h-1}, l_{h-1} + 1, \dots, l_h$ and $r, u = 1, 2, \dots, K$;

$y_{r,i;u,i+1}$: is equal to 1 if $t_{r,i} + d_i - a_i < t_{u,j} + b_{i+1}$; otherwise, it is equal to 0, where $i = l_1, l_2, \dots, l_h$.

4. Model the Revised Line

Assume that there was a virtual and additional tank $\tilde{p}$ performing S_{p+1}. Tank $\tilde{p}$ is placed between tank p and tank $p+1$, as shown in Figure 2. A revised scenario is formed, where the reentrance is relaxed compared to the original scenario shown in Figure 1. Moreover, constraints of the relaxed scenario, as shown in Figure 2, are necessary constraints of the original scenario shown in Figure 1. The operations in the revised scenario shown in Figure 2 should be analyzed and modeled first. Then, the operations concerning reentrance are analyzed, i.e. S_{p-1}, S_p and S_{p+1}. The revised scenario has been dealt with by Li and Fung (2013a). For completeness, the constraints of the revised scenario are formulated based on Li and Fung's (2013a) work. Simplified explanations are given as well.

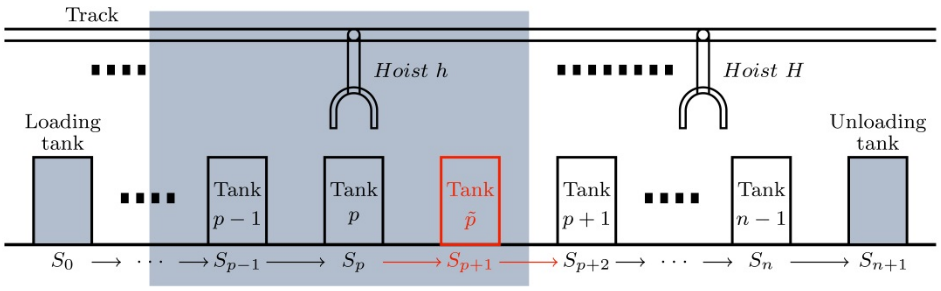

Figure 2: A Revised Multi-hoist Line without Reentrance

4.1 Constraints Concerning Hoist 1

4.1.1 Definitional and Initial Constraints

$$t_{\max 1} + \sum_{i=1}^{l_1} (d_i + e_{i+1,0}) x_i \le T \tag{1}$$

For $i = 0, 1, \dots, l_1$,

$$t_{\max 1} \ge t_{K,i} \tag{2}$$

For $i = 0, 1, \dots, l_1$,

$$t_{\max 1} \le t_{K,i} - (x_i - 1)M \tag{3}$$

$$\sum_{i=1}^{l_1} x_i = 1 \tag{4}$$

Constraints (2)-(4) guarantee that x_i and $t_{max\,1}$ are both well defined, i.e. $t_{max\,1}$ is the starting time of the last move performed by hoist 1. Constraint (1) guarantees that a cycle starting at hoist 1 picks up a part from stage 0. Hoist 1 comes back to stage 0 (i.e. loading tank) before the end of a cycle, for starting the next cycle.

4.1.2 Hoist Availability Constraints

For *r, u = 1, 2, ..., K*; *i < j* and *i, j = 0, 1,..., l*$_1$,

$$t_{u,j} - t_{r,i} \geq d_i + e_{i+1,j} - (1 - y_{r,i;u,j})M \quad (5)$$

$$t_{r,i} - t_{u,j} \geq d_j + e_{j+1,i} - y_{r,i;u,j}M \quad (6)$$

Constraints (5) (6) guarantee that the y's are well defined and that the hoist did not perform two moves simultaneously, i.e. moves performed by hoist 1 individually as a sequence.

4.2 Constraints Concerning Hoist h, Where h = 2, 3, …, H

4.2.1 Definitional and Initial Constraints

For $i = l_{h-1} + 1, ..., l_h$ and $h = 2, 3, …, H,$

$$t_{\max h} \geq t_{K,i} \quad (7)$$

For $i = l_{h-1} + 1, ..., l_h$ and $h = 2, 3, …, H,$

$$t_{\max h} \leq t_{K,i} - (z_i^h - 1)M \quad (8)$$

For $h = 2, 3, …, H,$

$$\sum_{i=l_{h-1}+1}^{l_h} z_i^h = 1 \quad (9)$$

For $j = l_{h-1} + 1, ..., l_h$ and $h = 2, 3, …, H,$

$$t_{\min h} \leq t_{1,j} \quad (10)$$

For $j = l_{h-1} + 1, ..., l_h$ and $h = 2, 3, …, H,$

$$t_{\min h} \geq t_{1,j} + (s_i^h - 1)M \quad (11)$$

For $h = 2, 3, …, H,$

$$\sum_{i=l_{h-1}+1}^{l_h} s_i^h = 1 \quad (12)$$

For $i = l_{h-1} + 1, ..., l_h$ and $h = 2, 3, …, H,$

$$\sum_{j=l_{h-1}+1}^{l_h} w_{i,j}^h - z_i^h = 0 \quad (13)$$

For $j = l_{h-1} + 1, ..., l_h$ and $h = 2, 3, …, H,$

$$\sum_{j=l_{h-1}+1}^{l_h} w_{i,j}^h - s_i^h = 0 \quad (14)$$

As constraints (2)-(4) are for hoist 1, constraints (7)-(9) guarantee that $t_{max\ h}$ and z_i^h are well defined. Similarly, constraints (7)-(9) guarantee that $t_{max\ h}$ and s_i^h are well defined. Constraints (13) and (14) make $w_{i,j}^h$ well defined.

4.2.2 Constraints for Cycle time T

For $h = 2, 3, …, H,$

$$T \geq t_{\max h} \quad (15)$$

$$T + t_{\min h} - t_{\max h} \geq \sum_{j=l_{h-1}+1}^{l_h} \sum_{i=l_{h-1}+1}^{l_h} (d_i + e_{i+1,j}) w_{i,j}^h \quad (16)$$

Constraint (15) means that each move should start during the cycle. Constraint (16) guarantees that there is enough time for hoist *h* moving to the stage that is concerned with the first move from the stage concerned with the end of the last move.

4.2.3 Hoist Availability Constraints

For *r, u* = 1, 2, …, *K* and $i = l_1, l_2, …., l_{H-1}$,

$$t_{u,j} - t_{r,i} \geq d_i + e_{i+1,j} - (1 - y_{r,i;u,j})M \quad (17)$$

$$t_{r,i} - t_{u,j} \geq d_j + e_{j+1,i} - y_{r,i;u,j}bM \quad (18)$$

Constraints (17) and (18) are similar to Constraints (5) and (6) concerning hoist 1.

4.3 Constraints to Avoid Conflicts between Hoists

For *r, u* = 1, 2, …, *K* and $i = l_1, l_2, …., l_{H-1}$,

$$t_{u,i+1} - (t_{r,i} + d_i) \geq (y_{r,i;u,i+1} - 1)M + \Delta \quad (19)$$

$$t_{r,i} + d_i - b_i - (t_{u,i+1} + a_{i+1}) \geq -y_{r,i;u,i+1}M \quad (20)$$

Constraints (19) and (20) make the y's well defined and guarantee that there are no conflicts concerning tank l_{h+1}, i.e. there are no conflicts between the hoists.

4.4 Tank Capacity Constraints

For $k = 1, 2, …, K$-1 and $i = 1, 2, …, n,$

$$y_{k,i-1;k,i} = y_{k+1,i-1;k+1,i} \quad (21)$$

$$y_{k,i-1;k+1,i} \geq 1 - y_{k,i-1;k,i} \quad (22)$$

$$y_{k+1,i-1;k,i} \leq 1 - y_{k,i-1;k,i} \quad (23)$$

In a line, move(*,i-1) inserts a part into stage i and move(*,i) picks up a part from stage i. Because stage i (with only one tank performing the stage) can process, at most, one part at a time, move(*,i-1) and move(*,i) must alternate with each other. Constraints (21)-(23) guarantee this condition.

4.5 Time Window Constraints

For $k = 1, 2, \ldots, K$-1 and $i = 1, 2, \ldots, n$,

$$t_{k,i} - (t_{k,i-1} + d_{i-1}) \geq L_i - (1 - y_{k,i-1;k,i})M \quad (24)$$

$$t_{k,i} - (t_{k,i-1} + d_{i-1}) \leq U_i + (1 - y_{k,i-1;k,i})M \quad (25)$$

$$t_{k+1,i} - (t_{k,i-1} + d_{i-1}) \geq L_i - y_{k,i-1;k,i}M \quad (26)$$

$$t_{k,i} - (t_{k,i-1} + d_{i-1}) \leq U_i + (1 - y_{k,i-1;k,i})M \quad (27)$$

If the stage is empty at the start of a cycle, parts are inserted and completed within the current cycle. Constraints (24) and (25) guarantee that time window constraints are satisfied in this case. If the stage is occupied at the start of a cycle, the part being processed is inserted during the previous cycle. The last inserted part during this current cycle will be processed at the end of the cycle. Constraints (28) and (29) correspond to the parts in this case. K-1 parts are inserted and completed during the cycle. Time window constraints concerning these K-1 parts are formulated by constraints (26) and (27).

4.6 Binary and Non-Negative Constraints

$$t_{1,0} = 0 \quad (30)$$

For $k = 1, 2, \ldots, K$-1 and $i = 0, 1, \ldots, n$,

$$t_{k,i} \geq 0 \quad (31)$$

For $k = 1, 2, \ldots, K$-1 and $i = 0, 1, \ldots, n$,

$$t_{k,i} + d_i \leq t_{k+1,i} \quad (32)$$

For $i = l_{h\text{-}1} + 1, \ldots, l_h$

$$x_i \in \{0,1\} \quad (33)$$

For $i = l_{h\text{-}1} + 1, \ldots, l_h$ and $h = 2, 3, \ldots, H,$

$$z_i^h \in \{0,1\} \quad (34)$$

For $j = l_{h\text{-}1} + 1, \ldots, l_h$ and $h = 2, 3, \ldots, H,$

$$s_j^h \in \{0,1\} \quad (35)$$

For $i, j = l_{h\text{-}1} + 1, \ldots, l_h$ and $h = 2, 3, \ldots, H,$

$$w_{i,j}^h \in \{0,1\} \quad (36)$$

For $r, u = 1, 2, \ldots, K$; $i < j$; $i, j = l_{h\text{-}1} + 1, \ldots, l_h$ and $h = 2, 3, \ldots, H,$

$$y_{r,i;u,j} \in \{0,1\} \quad (37)$$

For $r, u = 1, 2, \ldots, K$; $i = l_1, l_2, \ldots, l_{H\text{-}1},$

$$y_{r,i;u,i+1} \in \{0,1\} \quad (38)$$

5. Model Reentrance

As shown in Figure 1, stage p + 1 is performed in tank p – 1, where stage p -1 is also performed. When a part is completed in tank p for stage p, the part will be transported to tank p - 1 for stage p + 1. A conclusion of great significance is expressed by Lemma 1.

Lemma 1: It is infeasible that tank p - 1 and tank p are occupied at the same time.

Proof: Assume that part A and part B were being processed in tank p - 1 and p at the same time. Assume that part A was for stage p - 1. When part A is completed, it should be transported to tank p, which is occupied by part B. When part B is completed, it should be transported to tank p - 1 (for stage p + 1), which is occupied by part A. Hence, the deadlock occurs regardless of whether part A or part B is competed first.

Assume that part A is for stage p + 1. This means that part A entered the line before part B. When part A was transported from tank p to tank p-1 for stage p+1, tank p is empty. Moreover, tank p-1 started to process part A for stage p+1. It is infeasible to process part B at tank p - 1 for stage p - 1 at the same time and then to transport it to tank p.

Therefore, it is infeasible that tank p - 1 and tank p are occupied at the same time.

As a result, there are four feasible cases concerning Sp-1, Sp and Sp+1, based on the statuses of tanks p - 1 and p. These cases are listed as follows with corresponding move sequences.

Case (1): Both tanks p - 1 and p are empty at the start of a cycle.

The sequence of corresponding moves should be move(1, p -2) → move(1, p - 1) → move(1, p) → move(1; p + 1) → ⋯ → move(k, p -2) → move(k, p - 1) → move(k, p) → move(k, p + 1) → ⋯ → move(1, p -2) → move(1, p - 1) → move(1, p) → move(1; p + 1).

Case (2): Tank p - 1 is occupied for stage p - 1 and tank p is empty at the start of a cycle.

The sequence of corresponding moves should be move(1, p - 1) → move(1, p) → move(1; p + 1) →move(1, p -2) → ⋯ → move(k, p - 1) → move(k, p) → move(k, p + 1) → move(k, p -2) → ⋯ → move(1, p - 1) → move(1, p) → move(1; p + 1) → move(1, p -2).

Case (3): Tank p - 1 is occupied for stage p + 1 and tank p is empty at the start of a cycle.

The sequence of corresponding moves should be move(1, p) → move(1; p + 1) →move(1, p -2) → move(1, p - 1) → ⋯ → move(k, p) → move(k, p + 1) → move(k, p -2) → move(k, p - 1) → ⋯ → move(1, p) → move(1; p + 1) → move(1, p -2). → move(1, p - 1).

Case (4): Tank p - 1 is empty and tank p is occupied at the start of a cycle.

The sequence of corresponding moves should be move(1; p + 1) →move(1, p -2) → move(1, p - 1) →move(1, p) → ⋯ → move(k, p + 1) → move(k, p -2) → move(k, p - 1) →move(k, p) → ⋯ → move(1; p + 1) → move(1, p -2). → move(1, p - 1) → move(1, p).

Table 1 shows the expressions of four cases by y's. Therefore, one more constraint is required to guarantee that one of these cases should occur, namely Constraint (39).

For k = 1, 2, …, K,

$$y_{k,p-2;k,p-1} + y_{k,p-1;k,p} + y_{k,p;k,p+1} - y_{k,p-2;k,p+1} = 2 \quad (39)$$

Until now, the MILP model is complete. The objective is to minimize cycle time T subject to Constraints (1)-(39).

Table 1: Express These Cases by y’s

	yk,p-2;k,p-1	**yk,p-1;k,p**	**yk,p;k,p-+1**	**yk,p-2;k,p+1**
Case (1)	1	1	1	1
Case (2)	0	1	1	0
Case (3)	1	1	0	0
Case (4)	1	0	1	0

Table 2: Minimal and Maximal Processing Times (seconds)

Stage *i*	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Tank *i*	1	2	3	4	5	6	7	8	9	10	11	12
L_i	160	150	180	50	20	20	60	20	20	50	180	20
U_i	180	180	200	70	40	40	80	40	50	70	240	40
Stage *i*	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
Tank *i*	13	14	15	16	17	18	19	20	19	21	22	23
L_i	80	30	50	100	70	70	140	50	60	30	70	150
U_i	120	50	80	130	100	90	180	70	80	50	90	180

6. Illustrative Example

A numerical example is used to illustrate the MILP model proposed in this paper. The example is modified from Zhou and Li’s (2009) example. Using a personal computer with an i5-3337U 1.89GHz CPU and a 4GB RAM using the 64-bit Windows 8 OS, the commercial software CPLEX 12.6 under the default mode is used to solve the model with the example.

There are 2 hoists and 23 tanks performing 24 stages. Stage 19 and stage 21 are both performed in tank 19, i.e. $p = 20$. Moving times d_i = 14 seconds, where i = 0, 1, ..., 24. Empty moving time $e_{ij} = 6 + |i - j|$ seconds, where i, j = 0, 1, …, 25 and i ≠ j. $a_i = 3$ and $b_i = 4$, where i = 0, 1, …, 24. The processing times are listed in Table 2.

Tables 3, 4 and 5 list the results of the optimal schedules for 1-degree, 2-degree and 3-degree cycles respectively. Computation times are 1.52 CPUs, 130.52 CPUs and 1820.36 CPUs, respectively. Computation time around half an hour is viewed as an acceptable time, considering that the scheduling problem is off-line. Cycle times are 459s, 864s and 1259s respectively. Compared to 1-degree cycles, the improvements in the 2-degree cycles and 3-degree cycles are 3.79% and 6.53% respectively, i.e. (449-864/2)/449*100% = 3.79% and (449-1259/3)/449*100% = 6.53%. The results illustrate the benefits of multi-degree cycles.

Table 3: Result of 1-degree Cycle (l_1 = 12, T = 449s)

i	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
t_i	0	174	341	86	150	197	231	315	369	419	34	255	289
i		**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
t_i		405	0	64	182	266	350	87	151	226	290	374	111

Table 4: Result of 2-degree Cycle (l_1 = 12, T = 864s)

i	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
$t_{1,i}$	0	174	341	79	143	197	231	315	369	433	51	255	289
$t_{2,i}$	398	572	742	535	599	633	684	766	800	834	447	658	711
i		**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
$t_{1,i}$		419	483	64	178	262	346	136	211	305	369	453	110
$t_{2,i}$		811	864	547	671	755	839	507	571	645	696	780	621

Table 5: Result of 3-degree Cycle (l_1 = 12, T = 1259s)

i	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
$t_{1,i}$	0	174	361	59	143	197	251	335	389	453	31	225	276
$t_{2,i}$	309	483	647	555	619	669	703	785	834	885	527	760	809
$t_{3,i}$	730	914	1108	860	936	987	1038	1132	1179	1226	961	1155	1202
i		**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
$t_{1,i}$		52	96	160	274	27	131	325	409	459	560	0	194
$t_{2,i}$		383	436	523	637	358	496	613	690	764	828	664	858
$t_{3,i}$		943	1007	1078	1202	721	805	968	1037	1131	1175	912	1106

7. Conclusions

This paper considers multi-hoist scheduling with reentrance in multi-degree cycles. The no overlapping rule is applied and an MILP model is proposed based on this. The MILP model proposed is the first for this especially complicated scenario, i.e. multi-hoist lines with reentrance in multi-degree cycles. A numerical example is used to illustrate the model proposed. The results also show the benefits of multi-degree cycles.

This paper considers the special case of reentrance, i.e. stage p-1 and stage p+1 are performed in the same tank. For future work, a general case where stage q and stage v ($|v-q|>2$) are performed in the same tank will be investigated. Moreover, this paper applies the no overlapping rule to avoid conflict between the hoists. One direction is to pay more attention to the case where this rule is relaxed, which is a more general condition. Moreover, more attention will be paid to heuristic and/or meta-heuristic methods to deal with large-size instances.

Acknowledgement

The work described in this paper was supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. PolyU 510311); the NSFC (Project No. 71471158); and The Hong Kong Scholars Program Mainland–Hong Kong Joint Postdoctoral Fellows Program (Project No.: G-YZ24). The authors would also like to thank The Hong Kong Polytechnic University Research Committee for financial and technical support.

References

Che, A, Chabrol, M., Gourgand, M., & Wang, Y. (2012). Scheduling multiple robots in a no-wait re-entrant robotic flowshop. *International Journal of Production Economics, 135*(1), 199–208.

Che, A. & Chu, C. (2009). Multi-degree cyclic scheduling of a no-wait robotic cell with multiple robots. *European Journal of Operational Research*, 199(1), 77–88.

Che, A. Chu, C. & Chu, F. (2002) Multicyclic hoist scheduling with constant processing times. *IEEE Transactions on Robotics and Automation, 18*(1), 69–80.

Che, A., Lei, W., Feng, J., & Chu, C. (2014). An improved mixed integer programming approach for multi-hoist cyclic scheduling problem. *IEEE Transactions on Automation Science and Engineering, 11*(1), 302-309.

El Amraoui, A., Manier, M. –A., El Moudni, A., & Benrejeb, M. (2013a). A genetic algorithm approach for a single hoist scheduling problem with time windows constraints. Engineering Applications of Artificial Intelligence, 26(7), 1761–1771.

El Amraoui, A., Manier, M. –A., El Moudni, A., & Benrejeb, M. (2013b). A linear optimization approach to the heterogeneous r-cyclic hoist scheduling problem. *Computers & Industrial Engineering, 65*(3), 360-369.

Lei, W., A. Che, A., & Chu, C. (2014). Optimal cyclic scheduling of a robotic flowshop with multiple part types and flexible processing times. *European Journal of Industrial Engineering, 2*(8), 143–167.

Leung, J. M. Y., Zhang, G., Yang, X., Mak, R., & Lam, K. (2004). Optimal cyclic multi-hoist scheduling: a mixed integer programming approach. *Operations Research, 52*(6), 965-976.

Li, X., & Fung, R. Y. K. (2013a). A mixed integer linear programming approach for multi-degree cyclic multi-hoist scheduling problems without overlapping. In: *2013 IEEE International Conference on Automation Science and Engineering (CASE)*, Madison Wisconsin, USA, Aug. 17-21, pp. 274-279.

Li, X., & Fung, R. Y. K. (2013b). A mixed integer linear programming approach for multi-degree cyclic multi-hoist scheduling. *2013 Asia Pacific Industrial Engineering and Management Systems Conference*. Cebu, Philippines, Dec. 3-6, paper no. 1288.

Li, X., & Fung, R. Y. K. (2014). A mixed integer linear programming solution for single hoist multi-degree cyclic scheduling with reentrance. *Engineering Optimization, 46*(5), 704-723.

Liu, J., Jiang, Y., & Zhou, Z. (2002). Cyclic scheduling of a single hoist in extended electroplating lines: A comprehensive integer programming solution. *IIE Transactions, 34*(10), 905-914.

Lopez, P., & Roubellat, F. (2008). Production Scheduling. 1st (Ed.), *Chapter 8 Hoist Scheduling Problem*. London, UK: Wiley-ISTE.

Jiang, Y., & Liu, J. (2014). A new model and an efficient branch and bound solution for cyclic multi-hoist scheduling. *IIE Transactions, 46*(3), 249-262.

Manier, M. –A., & Bloch, C. (2003). A classification for hoist scheduling problems. *The International Journal* *of Flexible Manufacturing Systems, 15*(1), 37-55.

Phillips, L. W., & Unger, P. S. (1976). Mathematical programming solution of a hoist scheduling program. *AIIE Transactions, 8*(2), 219-225.

Shapiro, G. W., & Nuttle, H. L. W. (1988) Hoist scheduling for a PCB electroplating facility. *IIE Transactions, 20*(2), 157–167.

Steiner, G., & Xue, Z. (2005). Scheduling in reentrant robotic cells: Algorithms and complexity. *Journal of Scheduling, 8*(1), 25-48.

Zhou, Z., Che, A., & Yan, P. (2012). A mixed integer programming approach for multi-cyclic robotic flowshop scheduling with time window constraints. *Applied Mathematical Modelling, 36*(8), 3621-3629.

Zhou, Z., & Li, L. (2008). A heuristic algorithm for the two-hoist cyclic scheduling problem with overlapping hoist coverage ranges. *IIE Transactions, 40*(8), 782-794.

Zhou, Z., & Li, L. (2009). A solution for cyclic scheduling of multi-hoists without overlapping. *Annals of Operations Research, 168*(1), 5-21.

About Authors

Dr. Xin Li obtained a Ph.D. degree from Department of Systems Engineering and Engineering Management, at City University of Hong Kong in 2014. He received an M.S. degree in industrial engineering and logistics management from Shanghai Jiao Tong University, Shanghai, China, in 2011, and a B.S. degree in industrial engineering from Huazhong University of Science and Technology, Wuhan, China, in2007 respectively. His research interests include modeling, scheduling, and simulation of automated manufacturing systems using linear and discrete optimization methodology, and heuristics and evolution algorithms.

Prof. Felix T. S. Chan received his B.Sc. Degree in Mechanical Engineering from Brighton Polytechnic (now University), UK, and obtained his M.Sc. and Ph.D. in Manufacturing Engineering from the Imperial College of Science and Technology, University o London, UK. Prof. Chan is now working at the Department of Industrial and Systems Engineering, The Hong Kong Poly-technic University. His current re-search interests are Logistics and Supply Chain Management, Operations Management, Distribution Coordination, Systems Modelling and Simulation, Supplier Selection. To date, he has published 16 book chapters, over 300 refereed international journal papers and 250 peer reviewed international conference papers, h index = 30 under the Web of Science. He is a chartered member of the Chartered Institute of Logistics and Transport in Hong Kong.

Dr. Ben Niu received the B.S. degree from Hefei Union University, Hefei, China, in 2001, the M.S. degree from Anhui Agriculture University, Hefei, China, in 2004, and the PhD. degree from Shenyang Institute of Automation of the Chinese Academy of Sciences, Shenyang, China, in 2008. He is presently serving as an Associate Professor in Department of Management Science, Shenzhen University. Currently he is also a Postdoctoral Fellow at Hefei Institute of Intelligent Machines, CAS and at The Hong Kong Polytechnic University, respectively. His main fields of research are Swarm Intelligence, Bio-inspired Computing, and their applications on Supply Chain Optimization, Business Intelligence, and Portfolio Optimization.