# Three-tank hoist scheduling problem with unbounded or zero-width processing windows

Fabien Mangione, Nadia Brauner, Bernard Penz

# Three-tank hoist scheduling problem with unbounded or zero-width processing windows

**Fabien Mangione**
*GILCO, ENSGI-INPG, France.*
Fabien.Mangione@gilco.inpg.fr

**Nadia Brauner**
*Leibniz-IMAG, France.*
Nadia.Brauner@imag.fr

**Bernard Penz**
*GILCO, ENSGI-INPG, France.*
Bernard.Penz@gilco.inpg.fr

**Abstract**

On surface treatment lines, the products are transferred in successive tanks by a hoist. Moreover, each processing time is nested between a minimal and a maximal duration. This constraint is called the processing window constraint. An interesting problem is to find the hoist moves which maximise productivity. This paper deals with the production of identical parts in a balanced three-tank line. We prove that for zero-width or unbounded processing windows, production cycles of one or two parts are optimal.

## 1. Introduction

In surface treatment lines, products are immersed in several tanks. These tanks contain chemical baths like acids which affect the electrical or mechanical properties of the products. These kind of lines are used, for instance, for galvanoplasty or circuit board assembly. The products are mounted on carriers and transported from a tank to another one by a hoist. The time a part can stay in a tank is upper an lower bounded. The lower bound indicates the minimum time for a correct treatment. The upper limit is justified by the chemical properties of the baths. For instance, a product should not remain too long in an acid bath or you should not put too much precious metal on the products in order to minimise the costs.

A classical objective is to find the cyclic hoist moves which yield the maximum throughput. This problem is usually called Cyclic Hoist Scheduling Problem (CHSP). We shall restrict the problem to the production of identical parts. Lei and Wang [5] showed that the CHSP is NP-complete. Different methods were proposed to solve this problem: Constraint Logic Programming [2], Genetic Algorithm [6] or Branch and Bound Algorithm [8].

The problem with unbounded processing time windows (infinite upper bound) is usually called the robotic cell scheduling problem. It was introduced by Sethi *et al.* [9]. The zero-width time window problem (equal upper and lower bounds) with two and three machines was studied by Agnetis [1]. A survey on those problems was proposed by Crama *et al.* [4].

## 2. Problem description and notations

Figure 1 shows a surface treatment line with one hoist and $m$ treatment tanks. The carriers are moved from a bath to another one by a hoist which travels along a rail. Tank $0$ (the loading tank) and tank $m+1$ (the unloading tank) have infinite capacity whereas the other tanks of the line can only contain one carrier at a time.

The line represents a flow-shop with the hoist as the material handling device. All products are identical. A carrier is picked up at tank 0 and transferred in succession to tank 1, tank 2 etc. until it finally reaches the output station, tank *m+1*. The time the products remain in the baths is either fixed (zero-width processing windows) or lower bounded (infinite processing window). These hypotheses are close to industrial cases since for acids or precious metals the margin is almost zero and for rinsing tanks the products must stay a minimum time but can remain in the tank until the hoist is available.

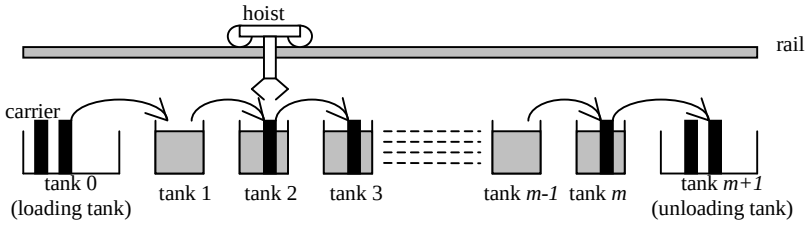


*Figure1*. A surface treatment line

The minimum processing time for a part in tank *i* is noted $p_i$. The effective time can be more if the time window is unbounded. We shall consider balanced line *i.e.* all minimum processing times are equal ($p_i=p$). The travel time of the hoist between two consecutive tanks is $\delta$. The travel times are additive. Hence, the hoist takes $|i-j|\delta$ time units to travel from tank *i* to tank *j*.

The hoist moves are described in terms of activities. Activity *Ai* consists of the following moves: the hoist picks up a carrier from tank *i*, then travels to tank *i+1* and loads the carrier onto tank *i+1*. The loading and unloading times can be neglected compared to the travel time.

We consider cyclic moves of the hoist and define the *k*-cycles. During the execution of a *k*-cycle exactly *k* carriers are treated and the state of the line is restored. Therefore all the activities *Ai* occur exactly *k* times.

The state of a system may be represented by a *m*-vector which *i*-th component is 0 if the tank *i* is empty and 1 otherwise. Define the state graphs *Gm* whose nodes are the states of the system and whose arcs represent the activities of the hoist to go from a state to another one [Figure 2].

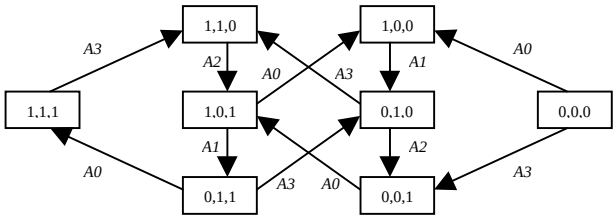


*Figure 2*. State graph for a three-tanks line, *G3*

Figure 3 describes the line-graph of the state graph of a 3-tank line. We can build the line-graph *LGm* of *Gm* as follows: the nodes of *LGm* are the arcs of *Gm;* (*a*,*a'*) is an arc of *LGm* if and only if there exists a node *v* in *Gm* which is the head of *a* and the tail of *a'*.

Define $E_3$, the set of nodes composing the cycle *(A0,A1,A2,A3)* (the grey nodes on Figure 3). Brauner and Finke. [3] proved the following property: For any instance *I*, let *l* be the smallest value such that there exists an optimal *l*-cycle $C_l$. Then $C_l$ crosses at most once a given node of $E_3$.

## 3. The three-tank case

In this section, we present an exhaustive study of the three-tank balanced hoist scheduling problem with identical products. We consider unbounded or zero-width processing windows on the tanks. For this problem we obtain the following result.

**Theorem.** For the identical part balanced three-tank hoist scheduling problem with unbounded or zero-width processing windows, the 1- and 2-cycles are dominant.

This theorem means that, in order to find the optimal production cycle in a three-tank line it is sufficient to consider only production cycles of one and two parts. An extension of this theorem to unbalanced lines can be found in [1] for the pattern (0,0,0) and in [4] for the pattern (∞,∞,∞).

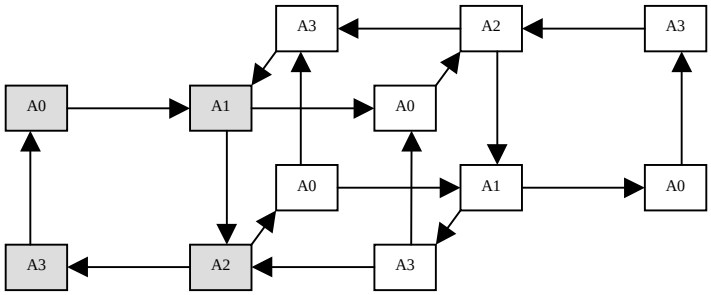


Figure 3. Line-Graph LG3 of G3

The optimal cycles are presented in Table 1. The columns indicate the links between $p$ and $\delta$. For instance in the first column one has $p \in [0,\delta[$. Each row represents a pattern of processing windows on the tanks. For instance, a pattern (0,∞,∞) indicates that the time a part can remain on the first tank is fixed (and equal to $p$) and that the processing times on tanks 2 and 3 are in the interval $[p,+\infty[$.

*Table 1*. Optimal cycles

<table>
<tr><th>p</th><th>[0,δ[</th><th>[δ,2δ[</th><th>[2δ,4δ[</th><th>[4δ,6δ[</th><th>[6δ,8δ[</th><th>≥ 8δ</th></tr>
<tr><td>0, 0, 0</td><td rowspan="8"></td><td rowspan="4">0, 1, 2, 3,</td><td rowspan="4"></td><td>0, 2, 1, 3, 2, 3,0, 1</td><td rowspan="3">0, 2, 1, 3, 2 , 0, 3, 1</td><td rowspan="7"></td></tr>
<tr><td>∞, 0, 0</td><td>0, 2, 3, 1</td></tr>
<tr><td>0, 0, ∞</td><td>0, 1, 3, 2</td></tr>
<tr><td>0, ∞, 0</td><td rowspan="3">0, 2, 1, 3</td><td rowspan="3">0, 3, 2, 1</td></tr>
<tr><td>0, ∞, ∞</td><td colspan="2">0, 1, 3, 2</td></tr>
<tr><td>∞, ∞, 0</td><td colspan="2">0, 2, 3, 1</td></tr>
<tr><td>∞, 0, ∞</td><td colspan="2"></td><td>0, 3, 1, 2</td><td colspan="2">0, 2, 1, 3, 2 , 0, 3, 1</td></tr>
<tr><td>∞, ∞, ∞</td><td colspan="2">0, 1, 3, 2 or 0, 2, 3, 1</td><td colspan="4"></td></tr>
</table>

For each pattern of processing windows, we proved optimality using the line graph *LG3* presented in Figure 3, or using the symmetry of patterns. In order to explain how it works, we give a proof for two cases with $p<4\delta$.

For the case (∞, 0, 0), in any admissible cycle, activity *A2* immediately follows *A1* and *A3* immediately follows *A2*. Indeed, the hoist does not have enough time to do other activities between *A1* and *A2* and between *A2* and *A3*. Once removed the unfeasible arcs from the line graph *LG3*, only remain the arcs between the nodes of $E_3$. Then the unique feasible cycle is the 1-cycle (*A0,A1,A2,A3*).

For the case (∞, 0, ∞), the constraints on the processing time in tank 2 implies that all feasible cycles contain the sequence (*A1,A2*). Therefore the arcs that start from a node *A1* must go to a node *A2*. Once removed all forbidden arcs and isolated nodes, one obtains the graph of Figure 4.

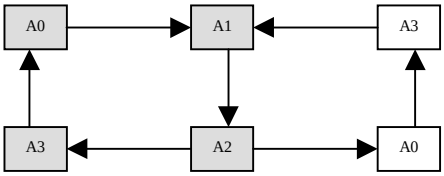


*Figure 4.* Line graph *LG3* for $p<4\delta$ and for the pattern (∞, 0, ∞)

Obviously, any *k*-cycle contains *k* activities *A1*. Hence any *k*-cycles with $k \geq 2$ crosses at least twice the node *A1* which is in $E_3$. Therefore the property given in section 2 indicates that, either the cycle (*A0,A1,A2,A3)* or the cycle (*A0,A3,A1,A2*) is optimal.

For $p > 8\delta$ the proof is straightforward. The cycle (*A0,A3,A2,A1*) is feasible for any pattern of processing windows. Crama *et al.* [4] showed that this cycle leads to a lower bound. Then it dominates all the other cycles.

For some cases, the line graph we obtain is more complex. Then we have to calculate and compare more precisely the throughput of the feasible cycles to prove that the 1- or the 2-cycles are dominant. The complete proof can be found in [7].

## 4. Concluding remarks and perspectives

In this paper, we studied a three-tank surface treatment line with one hoist as the material handling system. We proved that, for a balanced line with zero-width or unbounded processing windows, the production cycles of one and two parts are optimal. The next step will be to extend such results to larger lines (*m>3*) or to unbalanced three-tank lines.

## References


[1] Agnetis, A. (2000). Scheduling no-wait robotic cells with two and three machines, *European Journal of Operational Research*, 123 (2), 303-314.

[2] Baptiste, P., Legeard, B., Manier, M-A. and Varnier, C. (1996). Résolution d'un problème d'ordonnancement avec la PLC, *JESA*, 30 (2-3), 201-230.

[3] Brauner, N. and Finke, G. (2001). Cycles and permutations in robotic cells. *Mathematical and Computer Modelling*.34, 565-591.

[4] Crama, Y., Kats, V., van de Klundert, J. and Levner, E. (2000). Cycling Scheduling in Robotic Flowshop, *Annals of Operations Research: Mathematics of Industrial Systems*. 96, 97-124.

[5] Lei, L. and Wang, T.J. (1989). A proof: the cyclic HSP is NP-complete, *Technical Report 89-0016*, Graduate school of management, Rutgers Univ.

[6] Lim, J.M. (1997). A genetic algorithm for a hoist scheduling in the printed-circuit-board electroplating line, *Computers and Industrial Engineering*, 33 (3-4), 789-792.

[7] Mangione, F. and Brauner, N. and Penz, B. (2001). Balanced hoist scheduling problem with unbounded or zero-width processing windows, *Research Report*, GILCO.

[8] Ng, W.C. (1996). A Branch and Bound Algorithm for Hoist Scheduling of a circuit board production line, *International Journal of Flexible Manufacturing Systems*, 8, 45-65.

[9] Sethi, S.P., Sriskandarajah, C., Sorger, G., Blazewicz, J. and Kubiak, W. (1992). Sequencing of Parts and Robots Moves in a Robotic Cell, *The International Journal of Flexible Manufacturing Systems*, 4, 331-358.