Resolution of graphs with Bounded Cycle Time for the Cyclic Hoist Scheduling Problem

Manuel Mateo, Ramón Companys, Joaquín Bautista
Dpto. de Organización de Empresas, Polytechnic University of Catalonia, Spain.
mmateo@oe.upc.es, companys@oe.upc.es, bautista@oe.upc.es

Abstract

We deal with the *Hoist Scheduling Problem*, where a hoist carrying products along a production line must be scheduled. Products are taken from a bath once finished an operation and moved to the following one. There are time windows (a minimum and a maximum values) for the time spend at each treatment. The objective is to minimise the total production time. Our contribution is the use of bounds on the variable to be optimised, the Cycle Time. These bounds are obtained from the graph with Bounded Cycle Time. A resolution based on this oriented graph can be justified using some properties of dynamic programming.

1. Introduction

Some production lines are formed by a set of tanks or baths containing chemical treatments, without buffers between them. Due to the chemical nature, processing times must last within specific ranges of values, i.e. between a minimum and a maximum duration. So, if any time constraint is surpassed, the product will be considered as defective. The objective is to maximise the production rate meeting the whole set of constraints.

Hoists are quite frequent in electroplating and PCB (Printed Circuit Boards) production, to move materials between tanks. Scheduling and control of such hoists becomes a key factor to the proper work of the system, specially since chemical processes are involved. Control of the hoist movements is known as *Hoist Scheduling Problem* (HSP) [6]. HSP, even in the simplest case with one hoist and a kind of product, is NP-complete, as it was proved by Lei and Wang [3].

The problem to schedule one hoist in a line has the following characteristics:

- Each task has a specific time window (a maximum duration minus a minimum duration).
- Hoist moves products between two tanks, carrying at most one at a time. Trip, loading and unloading times are considered constant.
- Each tank has capacity for one product. There are no buffers between two adjacent stations.

When batch sizes are quite large, the same scheduling remains for a long time, which assumes that hoists repeat indefinitely a sequence of movements, known as cyclic sequence.

Given these constraints, the objective is to determine in which instant products must be taken from tanks, trying to maximise production, i.e. the hoist is scheduled to minimise the cycle time (CT) for the simplest case, with homogenous products. This work uses bounded values in graphs to solve the *Cyclic Hoist Scheduling Problem* (CHSP) [2].

2. A graph G, with Bounded Cycle Time, for the CHSP (Cyclic Hoist Scheduling Problem)

Phillips and Unger [6] presented the first model for CHSP by means of MILP. Shapiro and Nuttle [7] adapted it for a branch-and-bound algorithm. According to a hoist movement sequence, Mateo, Companys and Bautista [5] defined the problem as follows:

$$[\text{MIN}]\ CT \qquad (1)$$

$$\text{s.t. } t_{Gto(j)} - t_{Gfrom(j)} \geq Gtime(j) + Gcycle(j) \cdot CT \qquad j=1, ..., 3m+1 \qquad (2)$$

$$t_i \geq 0 \qquad i=0, ..., m \qquad (3)$$

$$CT \geq 0 \qquad (4)$$

with Gfrom, Gto $\in$ {0, 1, ..., m}; Gtime $\in$ Z (positive or negative); Gcycle $\in$ {-1,0,1}.

Looking at the constraints, Mateo [4] suggests the search for the maximum path in a graph **G**. A vertex v(i) is associated to each variable t_i (i=0, ..., m), and an arc $\mathbf{e[v(t_{Gfrom}),v(t_{Gto})]}$ from vertex $v(t_{Gfrom})$ to vertex $v(t_{Gto})$ with value:

$$b_j(CT) = Gtime(j) + Gcycle(j) \cdot CT \qquad (5)$$

is equivalent to the constraint (2).

Considering two groups of arcs [4], some properties in the graph **G** can be established:

P1. The graph **G** has *m+1* vertices: v(0), v(1), ..., v(m).

P2. The vertices v(0) and v(m) in the graph **G** receive two incident arcs: a "tank arc" and a "hoist arc", as it can be proved with the list of constraints.

P3. The other vertices in the graph **G**, [v(1), ..., v(m-1)], receive three incident arcs: two "tank arcs" and a "hoist arc", as the list of constraints shows.

P4. The total amount of arcs in the graph **G** is *3m+1*, associated to the *3m+1* constraints in the problem.

P5. Depending on the assigned value to CT, the graph **G** may have bounded or unbounded paths.

We say a graph **G** is **coherent** with CT=C, what is shown as **Coh[CT=C]**, if a bounded path exists for any pair of vertices [v(i), v(j)]. In the graph **G**, there are $3^{m-1} \cdot 2^2$ subgraphs $\mathbf{G_l}$, each one of which is formed by a set of incident arcs on each one of the *m+1* vertices in that graph. A lower bound and an upper bound for CT can be obtained by means of a selection of arcs in the graph **G**, all of them in a circuit, and one of them, at least, with length depending on CT.

3. Solving the graph G with Bounded Cycle Time

3.1. From the graph G to a matrix form

Let a vector **X** be composed by a set of components associated to the vertices of the graph **G**. Let be F_I a transformation from a vector **X** into another vector **Y** (with identical properties than **X**) such that $\mathbf{X,Y} \in \Re^{m+1}$, consisting in the application of:

$$y_i = \mathrm{MAX}_{k(i)} \{ b_{j,i}^{k(i)} + x_j^{k(i)} \} \qquad i = 0, ..., m \qquad (6)$$

with $b_{j,i}^{k(i)}(CT)$; i,j$\in$ {0,1, ..., m}; k(i)$\in$ {1,2} for i=0,m; k(i)$\in$ {1,2,3} for i=1,...,m-1.

Proposition 1

A transformation F_I can be associated to any graph **G**, which is called graph with Bounded Cycle Time (graph BCT), because bounds on CT are useful for the convergence.

Proposition 2

The transformation F_I splits $\Re^{m+1}$ into areas according to the active constraints at each vertex *i*, being $3^{m-1} \cdot 2^2$ the maximum number of areas.

A reduced transformation $F_{I,l}$ corresponds to each area *l*, and a subgraph $\mathbf{G_l}$ can also be associated to such area, in which only one arc is incident on each vertex of that subgraph.

Proposition 3

In a subgraph $\mathbf{G_l}$ there is at least a circuit with *k* arcs (k≤m+1).

Considering the above Proposition 3, if $\mathbf{X} \in \Re^{m+1}$ is inside an area *l*, the reduced transformation $F_{I,1}$ is equivalent to a linear system $\mathbf{Y} = \mathbf{B} + \mathbf{A} \cdot \mathbf{X}$, where each row corresponds to the active constraint incident to each vertex.

3.2. Double points in the application $F_{I,1}$

The transformation $F_{I,1}$ has a double point $\mathbf{X^o}$, at least, if $\mathbf{X^o} = F_{I,1}(\mathbf{X^o})$ is accomplished.

Double points indicate the repetition of the values for the *m+1* vertices in the subgraph $\mathbf{G_l}$ after a period CT. Nevertheless, existence of double points depends on values $b_{j,i}^{k(i)}$.

Theorem 1

If a double point $\mathbf{X^o}$ exists for $F_{I,l}$, any circuit in the graph $\mathbf{G_l}$ has null value.

Corollary

If all the circuits in graph $\mathbf{G_l}$ have null value, double points exist for $F_{I,l}$.

Therefore, values of CT for coherent graphs lead to null value for any circuit.

3.3. Determining double points in the application $F_{I,1}$

Let be $\mathbf{X^o}$, $\mathbf{X^1}$ such that $\mathbf{X^1} \neq \mathbf{X^o}$, which also accomplish the expression $\mathbf{X} = \mathbf{B} + \mathbf{A} \cdot \mathbf{X}$. Then:

$$(\mathbf{X^0} - \mathbf{X^1}) = \mathbf{A} \cdot (\mathbf{X^0} - \mathbf{X^1}) \tag{7}$$

If $F_{I,l}$ has double point $\mathbf{X^o}$, according to Definition 4, it is convenient to study what happens if $\mathbf{X^1} \neq \mathbf{X^0} + k \cdot \mathbf{1}\ \forall k$, supposing a single circuit, i.e. $\mathbf{Y^1} = F_{I,1}(\mathbf{X^1})$ with $\mathbf{Y^1} \neq \mathbf{X^1}$.

Proposition 4

If $\mathbf{X^1}$ is not a double point of $F_{I,1}$, for p great enough, then:

$$F_{I,1}^{p+q}(\mathbf{X^1}) = F_{I,1}^{p}(\mathbf{X^1}) \tag{8}$$

with q multiple of the number of vertices in the circuit, and in case of several circuits, power of the minimum common multiple for the respective amounts of vertices.

Proposition 5

In the above conditions, with $\mathbf{X^1}$ defined as in Proposition 6 and $\mathbf{X^2}$ such that $\mathbf{X^2} = F_{I,1}(\mathbf{X^2})$:

$$\mathbf{X^2} = \frac{1}{q} \sum_{j=1}^{q} f_{I,l}^{\,p+j}(\mathbf{X^1}) \tag{9}$$

Given these propositions, it can be observed that:

1. If $\mathbf{C}$ is the set of vertices in a circuit of value s, then:

$$\sum_{i \in \mathbf{C}} x_i + s = \sum_{i \in \mathbf{C}} y_i \quad \text{with } \mathbf{Y} = \Phi_{I,1}(\mathbf{X}) \tag{10}$$

2. To stabilise recurrence, it can be necessary to normalise the vector $\mathbf{Y}$:

$$\mathbf{Y'} = \mathbf{Y} - (s/p) \cdot \mathbf{1} \tag{11}$$

where p is the number of vertices in a circuit.

3. In the proposed procedure, following the scheme of dynamic programming [1], we are interested in $\mathbf{g}$=0 as our double point is $\mathbf{W}$:

$$\mathbf{B} + \mathbf{A} \cdot \mathbf{W} = \mathbf{g} + \mathbf{W} \tag{12}$$

3.4. Oscillation with period *p* around points in the application $F_{I,1}$

Theorem 2

If all the circuits in a graph have no positive value, a convergent algorithm ends in a finite number of steps, according to ε, giving a vector $\mathbf{X}$ with bounded distance to the double point:

$$|\,\mathbf{X} - (\mathbf{B} + \mathbf{A} \cdot \mathbf{X})\,| < \varepsilon\, [1,1,...,1]^T \tag{13}$$

If any circuit has positive value, one must dealt with the convex components one to one. For each convex component, another algorithm with $g = y_0$ and $y_i = y_i - g$ (i =0, ..., m) can be applied.

Theorem 3

If the subgraph $\mathbf{G_l}$ has a single convex component, or equivalently class, this second algorithm ends in a finite number of steps, according to ε.

Corollary

If $|g| > \varepsilon$, then the circuit has no null value.

4. Computational results

An algorithm, similar to that in [7] and considering this graph, has been applied to 540 instances of CHSP, with a number of baths ranging from 5 to 10. The instances are classified according to two data parameters: the width of time windows and the hoist speed.

Minimum time spent at bath *i*, a_i, is generated using a uniform distribution from 20 to 80. Maximum time spent at bath *i*, b_i, is generated according to the corresponding kind of windows: close windows (CW), $b_i = U[1.2a_i, 1.5a_i]$; half-opened windows (HW), $b_i = U[1.5a_i, 2a_i]$; and open windows (OW), $b_i = U[2a_i, 10a_i]$. On the other hand, the duration of hoist movements without load between consecutive baths is generated with a distribution $e_{i,i+1} = U[5,10]$. The hoist can be seen as: fast hoist (FH), $f_i = 1.5e_{i,i+1}$; half-fast hoist (HH), $f_i = 2e_{i,i+1}$; and slow hoist (SH), $f_i = 3e_{i,i+1}$. Combining a kind of windows with a kind of hoist, there are 9 sets with 60 instances each one (10 instances per number of baths *m*, $5 \le m \le 10$).

Table 1 shows the mean percentage of rejected sequences after the called Test of Unfeasibility, i.e. the relation between solved and planned graphs applying the definition for a **coherent** graph **G**. To search for the optimal schedule, a branch and bound algorithm is used. For each hoist sequence or subsequence, an algorithm based on the evaluation of vertices on graph BCT is solved. "Rejected sequences" is used for complete sequences and also for subsequences.

Each file corresponds to the results for instances with that number of baths. The three first columns show results according to the time windows; three next columns are referred to the hoist speed; and, the last one, is the mean result for the 90 instances with that number of baths.

Table 1. Percentage of rejected sequences after Test of Unfeasibility.

m	CW	HW	OW	FH	HH	SH	all
5	0.13	0.08	0.09	0.12	0.10	0.08	0.10
6	0.14	0.11	0.05	0.08	0.12	0.10	0.10
7	0.14	0.10	0.08	0.10	0.13	0.10	0.11
8	0.17	0.14	0.10	0.12	0.13	0.17	0.14
9	0.17	0.15	0.14	0.16	0.14	0.15	0.15
10	0.19	0.16	0.16	0.16	0.17	0.18	0.17

The results show the number of rejected sequences for hoist movements in the Test of Unfeasibility, which varies between 10% and 20% for lines with this number of stations.

Acknowledgements

This work has been supported by TAP98-0494 project.

References

[1] Bellman, R. (1952). On the theory of dynamic programming, *Proceedings of the National Academy of Sciences*, 38, 716-719.

[2] Chen, H., Chu, C. and Proth, J.-M. (1995). Cyclic hoist scheduling based on graph theory *IEEE 0-7803-2535-4/95*, 451-459.

[3] Lei, L. and Wang, T.-J. (1989). A proof: the cyclic hoist scheduling problem is NP-complete, *Working paper #89-0016*, Rutgers University.

[4] Mateo, M. (2001). *Procedimientos de secuenciación y programación en un sistema productivo de estaciones en serie con transportadores asíncronos de material*, PhD thesis, Universitat Politècnica de Catalunya, Barcelona.

[5] Mateo, M., Companys, R. and Bautista, J. (2000). Bounded Cycle Time for the Cyclic Hoist Scheduling Problem, *I World Conference on Production Operations Management,* Sevilla.

[6] Phillips, L.W. and Unger, P.S. (1976). Mathematical programming solution of a hoist scheduling program, *AIIE Transactions*, vol. 8, nº 2, 219-225.

[7] Shapiro, G.W. and Nuttle, H.W. (1988). Hoist Scheduling for a PCB electroplating facility, *IEE Transactions*, vol. 20, nº 2, 157-167.