Ordonnancement des cellules robotisées pour une production mono-produit

Fabien Mangione, Nadia Brauner, Bernard Penz

Cyclic Robotic Cells Scheduling for an Identical Part Production

Fabien Mangione[1], Agustìn Pecorari[1], and Bernard Penz[1]

GILCO laboratory, Grenoble, France
e-mail: mangione,penz@gilco.inpg.fr

Keywords. Flow-shop, scheduling, material handling system, cyclic production, state graph.

1 Introduction

In robotic cells, the products are mounted on carriers and transported between two machines by a robot. The problem is usually called the robotic cell scheduling problem. It was introduced by Sethi et al (1992). A survey on those problems was proposed by Crama et al. (1997). The paper is organised as follows. The description and the notations of the problem are given in the next section. The following section deals with the cycle time calculation problem. Methods to build the cycles are given in section 4 and the last section concludes this study.

2 Problem description and notations

The m machines of the line are denoted by M_1, $M_2 \ldots M_m$ [Figure 1]. The loading machine, M_0, and the unloading machine, M_{m+1}, have infinite capacity whereas the other machines of the line can only contain one carrier at a time. The robot takes δ time units to travel between two consecutive machines. Thus the travel time from M_i to M_j is $|i-j|\delta$ time units.

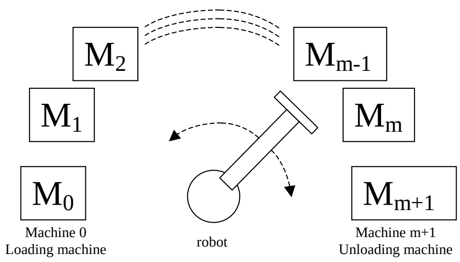

Fig. 1. Robotic Cells

The line represents a flow-shop with the hoist as the material handling device. All products are identical. A carrier is picked up at M_0 and transferred in succession to M_1, M_2 etc. until it finally reaches the output station, M_{m+1}. The time the products remain in the baths is unbounded. The minimum time a part has to remain in machine M_i is denoted by p_i.

The hoist moves are described in term of activities. Activity i ($i = 0, 1 \ldots m$) consists in the following moves:

- the hoist picks up a carrier from M_i;
- the hoist travels from M_i to M_{i+1};
- the hoist loads the carrier onto M_{i+1}.

We consider cyclic moves of the hoist and define the k-cycles in terms of activities. During the execution of a k-cycle (k degree cycle), exactly k carriers are treated and the state of the line is restored. Therefore all the activities A_i occur exactly k times.

The state of a system may be represented by a m-vector which i$^{\text{th}}$ component is 0 if the tank i is empty and 1 otherwise. Define the state graphs where the nodes are the state of the system and the arcs represent the activities of the hoist to go from a state to another one [Figure 2].

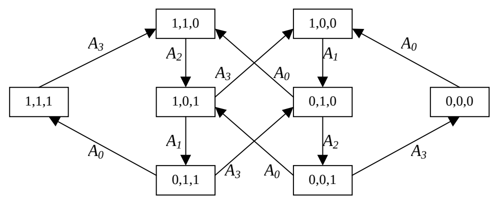

Fig. 2. State graph for a three tanks line: $S3$

The relation between a feasible cycle and the state graphs are the following: each feasible cycle can be represented by a circuit in the state graph and each circuit in the state graph represents a feasible cycle.

3 Cycle time

The calculation of the cycle time can be divided in two parts. The first part is to calculate the duration of the robot moves and the second part is to calculate the waiting times of the robot.

The first part can be done by a simple algorithm in $O(k(m+1))$ where k is the degree of the cycle and m is the number of machines. This algorithm sums the robot moves while the robot transports the parts and when it travels without. For the second part, the calculation of the waiting times can be done by a linear program. For example for the cycle A_0, A_2, A_4, A_1, A_3 and the processing times [12;12;12;10]. The linear program is the following one (t_i is the waiting time in machine i):

$$\begin{cases} t_2 = \max(0; 4 - t_3) \\ t_4 = \max(0; 2 - t_2) \\ t_1 = \max(0; 4 - t_2 - t_4) \\ t_3 = \max(0; 4 - t_4 - t_1) \end{cases} \Longleftrightarrow \begin{cases} t_2 + t_3 \geq 4 \\ t_2 + t_4 \geq 2 \\ t_2 + t_4 + t_1 \geq 4 \\ t_4 + t_1 + t_3 \geq 4 \\ t_2, t_4, t_1, t_3 \geq 0 \\ Min : z' = t_2 + t_4 + t_1 + t_3 \end{cases}$$

This program can be use for any cycle degree. Moreover we proved that the waiting times we obtain are the minimum mean waiting times.

4 Cycle Construction

The objectif of the robotic cells scheduling problem is to find the optimal cycle. This section section deals with an enumerative method which gives all the feasible cycles then comparing the cycle times gives the optimal cycle.

4.1 Properties of the state graphs

The following properties can easily be proved:

- The outcome degree in the state graph is bounded as follows:
  $1 \leq deg(v)^{+} \leq \lfloor \frac{m+2}{2} \rfloor$,
- The k-cycle lengths are $m+1$ multiple,
- The number of activity A_0 in the state graph is equal to 2^{m-1}
- The number of activities in the state graph is equal to $(m-1).2^{m-2} + 2.2^{m-1}$

4.2 Construction

The relation between the cycles and the state graph can be used to find all the feasible cycles. Indeed to find all the cycles we just have to find all the circuit in the state graph.

First we used an algorithm based on a backtrack procedure in tree which begins to search from activity A_0 (algo. # 1). Then we used the following property: if one searches from a state with π_0 products in the line, the number of products in the line cannot be more than $\pi_0 + k$ and less than $\pi_0 - k$. This property reduce the number of possible node in the graph (algo. # 2). Next, during the construction of a cycle, we delete all the arcs already crossed which prevent from finding a same cycle several times (algo. # 3). Finally we calculate the number of activities needs to restore the initial state of the line and then, remove some nodes from the graph (algo. # 4).

The table 1 gives the number of k-cycles with $k \leq m-1$, and m for 2 to 5. The number of cycles can be compared with the optimal number of cycles gives by Brauner (1999). Even if we still find more than the optimal number the algorithms give good results

m	algo. # 1	algo. # 2	algo. # 3	algo. # 4	optimal
2	2	2	2	2	2
3	40	40	28	28	26
4	11440	11440	4952	4952	3940
5	118017288	118017288	35640478	35640478	29604116

Table 1. Number of cycles

The table 2 gives the computation time (in seconds) for the preceeding calculation.

If we combine this method and the calculation of the cycle time we can find the optimal cycle for every processing times.

m	algo. # 1	algo. # 2	algo. # 3	algo. # 4
2	0,00	0,00	0,00	0,00
3	0,01	0,00	0,01	0,00
4	0,16	0,18	0,08	0,06
5	4254,07	4655,51	1404,06	807,19

Table 2. Computation time (s)

5 Conclusions and perspectives

In this paper we studied the cyclic robotic cells problem. We proposed a method to calculate the cycle time and we proposed a method to find all the feasible cycles. Mixing those two results we proposed a method to find the optimal cycle for small instances. The next step will be to find a way to calculate the cycle time during the cycle construction. This method would give bounds which can limit the calculation. The current direction is to study the "*hoist scheduling problem*". In this problem, the processing times are bounded which decrease the number of feasible arcs in the state graph.

References

BRAUNER-VETTIER N. (1999): Ordonnancement dans des cellules robotisées. *Phd report*. Universit Joseph Fourier, Grenoble, France.

CRAMA, Y. and VAN DE KLUNDERT, J. (1997): Cyclic scheduling of identical parts in a robotic cell. *Operation Research*, *45*, 6, 952–965.

CRAMA, Y. and KATS, V. and VAN DE KLUNDERT, J. and LEVNER, E. (2000): Cyclic scheduling in a robotic flowshops. *Annals of Operation Research : Mathematics of Industrial Systems*, *96*, 97–124.

SETHI S.P. and SRISKANDARAJAH C. and SORGER G. and BLAZEWICZ J. and KUBIAK W. (1992): Sequencing of Parts and Robots Moves in a Robotic Cell. International Journal of Flexible Manufacturing Systems, *4*, 331-358.