Discrete Applied Mathematics 156 (2008) 2480–2492

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

# Identical part production in cyclic robotic cells: Concepts, overview and open questions

Nadia Brauner

*G-SCOP, UJF, 46 avenue Felix Viallet, 38031 Grenoble Cedex, France*

ARTICLE INFO

*Article history:*
Received 18 September 2006
Received in revised form 27 June 2007
Accepted 6 March 2008
Available online 25 April 2008

*Keywords:*
Scheduling
Flow-shop
Material handling system
Cyclic production

ABSTRACT

Robotic cells consist of a flow-shop with a robot for material handling. A single part is to be produced cyclically and the objective is to minimize production rate. This document introduces basic concepts and tools for dealing with cyclic production. In particular, it concentrates on $k$-cycles which are production cycles where exactly $k$ parts enter and leave the cell. One defines the cycle function $\mathcal{K}$ which is the smallest value of $k$ so that the set of all $k$-cycles up to size $\mathcal{K}$ contains an optimal cycle for all instances. Known results and conjectures on these functions are given for the classical case where parts can remain on the machine waiting for the robot and for the no-wait case where parts have to be removed from the machine as soon as their processing is finished.



Robotic flow-shops consist of $m$ machines served by a single central robot. They were first introduced by [4] and studied by [40]. In [4], a line for machining castings for truck differential assemblies is described in the form of a 3-machine robotic cell where a robot has to transfer heavy mechanical parts between large machines. The system contains a conveyor belt for incoming parts and another one for outgoing parts. In this particular system the robot is not able to traverse the conveyor. Therefore, the movement of the robot from the output to the input station has to traverse the entire cell.

The original application has the form of a flow-shop. However, robotic cells may have very flexible configurations. The robot can easily access the machines thus producing a large variety of products in form of a job-shop. But it is known that the robotic scheduling problem is already NP-hard for a flow-shop with $m \geq 3$ machines and two or more different part types [28]. The case of the $m$-machine robotic cell in which one wants to produce a single batch of identical parts remains of interest. We shall mainly concentrate on this case. The robot may have unit capacity, as will be the case in our model, or one may have two-unit robots [42,41] or a multi-robot cell [32,30]. Robotic cells with buffers at the machine have been studied in [24,13]. Operation and/or process flexibility (the order of the operations or the assignment of the operations to the machines are not fixed) was recently studied in [3,27,26]. We concentrate here on the no-buffer case where parts are either on a machine or transported by the robot and no flexibility on operations or on the process is allowed. A survey on general robotic cells can be found in [18,23].

This document gives a state of the art on cyclic scheduling of identical parts in robotic cells. It describes classical configurations of robotic cells (Section 1) and introduces basic concepts and tools for dealing with cyclic production (Section 2). Then it describes known results for the classical case where parts can remain on the machine waiting for the robot (Section 3) and for the no-wait case where the parts have to be removed from the machine as soon as their processing is finished (Section 4). Then a detailed list of open questions is given (Section 5).

## 1. Robotic cells

The $m$ machines of a robotic cell are denoted by $M_1, M_2 \ldots M_m$ and we add two auxiliary machines, $M_0$ for the input station IN and $M_{m+1}$ for the output station OUT (Fig. 1). Raw material for the parts to be produced is available in unlimited quantity

*E-mail address:* nadia.brauner@g-scop.inpg.fr.

0166-218X/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2008.03.021

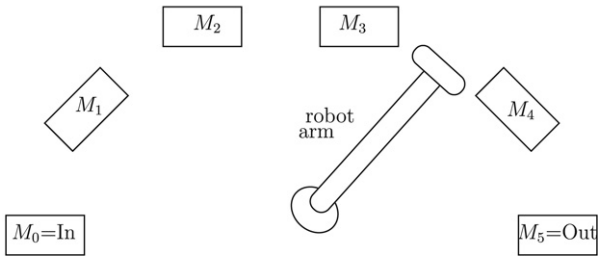


**Fig. 1.** Robotic cell with $m = 4$ machines.

**Table 1**
Two different expressions of the production constraints

| | Processing | Waiting policy | Another description |
|---|---|---|---|
| No-wait | $p_i$ | 0 | $\underline{p_{ij}} = \overline{p_{ij}}$ |
| HSP | in $[\underline{p_{ij}}, \overline{p_{ij}}]$ | 0 | $\underline{p_{ij}} \leq \overline{p_{ij}}$ |
| Unbounded | $p_i$ | Unbounded | $\overline{p_{ij}} = +\infty$ |

at $M_0$. The central robot can handle a single unit at a time. A part is picked up at $M_0$ and transferred in succession to $M_1$, $M_2 \ldots M_m$, where it is machined in this order until it finally reaches the output station $M_{m+1}$. At $M_{m+1}$, the finished parts can be stored in unlimited amounts. We focus on the classical case as in [40], where machines $M_1, M_2 \ldots M_m$ are without buffer facility. In this case, the robot, with unit capacity, has to be empty whenever it wants to pick up a part at $M_h$ ($h = 0, 1 \ldots m$).

Consider an instance $I$ of an $m$-machine robotic cell. Different cell configurations have been studied, depending mainly on production constraints (Section 1.1) and on the metric for travel times (Section 1.2).

### 1.1. The production constraints

Processing starts as soon as a part is loaded on a machine. The *processing time* represents the minimum time a part must remain on a machine. If all parts are different, $p_{ij}$ denotes the processing time of part $j$ on machine $M_i$. If one wants to produce one large batch of *identical parts*, the processing times of the parts on machine $M_i$ are $p_i = p_{ij}$. In the *balanced* case, all processing times are equal, *i.e.*, $p_i = p$ for all $i$.

Once the part is finished, two policies may apply. In the *no-wait* case, the part must be removed immediately from the machine and transferred to the following machine. In the *unbounded* case, the part can remain on the machine waiting for the robot.

A classical extension of those two cases is the so-called Hoist Scheduling Problem (HSP) for which the processing policy is different. For the preceding two cases, the processing time is fixed. For the HSP, the processing time is described by an interval, and the no-wait policy applies. This means that the time part $j$ may remain on machine $M_i$ lays in the interval $[\underline{p_{ij}}, \overline{p_{ij}}]$ (see Table 1). This applies to chemical treatments were the machines correspond to chemical baths. The HSP is NP-hard even for identical parts and very simple (additive) configurations of cells [20]. There exists a wide literature on this problem that we will not develop here (see e.g. [5]).

Denote by $\epsilon$ the time to load a part onto a machine from the robot or to unload a part from a machine onto the robot.

### 1.2. The travel metric of the robot

We shall consider different classical metrics for travel times of the robot depending on the physical configuration of the cell and on the characteristics of the robot. Denote by $\delta_{h,h'}$, the travel time of the robot (empty or loaded) from $M_h$ to $M_{h'}$. The following natural, and in practice desirable, assumptions are made [14]:

- the travel time from a machine to itself is zero, that is, $\delta_{h,h} = 0$;
- the travel times satisfy the triangle inequality, that is, $\delta_{h,k} + \delta_{k,h'} \geq \delta_{h,h'}$ for all $h$, $k$ and $h'$;
- The travel times are symmetric, that is $\delta_{h,h'} = \delta_{h',h}$ for all $h$ and $h'$.

Travel times verifying those three assumptions are called *general* (or sometimes "euclidean", e.g. in [23]). Special configurations of cells have been studied. Table 2 summarizes the most classical ones which we now define in detail.

For *additive* times, to travel between distant machines, the robot passes through all intermediate machines and its speed is constant. This metric is the most popular since, in practice, it is applicable if the machines are on a circle or on a line and if the cell is dense (the robot does not have time to speed up between distant machines). In this case, one has the triangle equality $\delta_{h,h'} = \delta_{h,k} + \delta_{k,h'} = \sum_{k=h}^{h'-1} \delta_{k,k+1}$ for any $h < h'$. By symmetry, this also defines $\delta_{h,h'}$ for $h > h'$. Some authors