Cyclic Job Shop Problem with varying processing times
Idir Hamaz, Laurent Houssin, Sonia Cafieri

To cite this version:
Idir Hamaz, Laurent Houssin, Sonia Cafieri. Cyclic Job Shop Problem with varying processing times.
IFAC-PapersOnLine, 2017, 50 (1), pp.5012 - 5016. <10.1016/j.ifacol.2017.08.908>. <hal-01626534>

HAL Id: hal-01626534
https://hal-enac.archives-ouvertes.fr/hal-01626534
Submitted on 30 Oct 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Cyclic Job Shop Problem with varying processing times

Idir Hamaz *,**, Laurent Houssin ***, Sonia Cafieri ***

* CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France
(e-mail: idir.hamaz@laas.fr)

** Univ de Toulouse, UPS, LAAS, F-31400 Toulouse, France

*** ENAC, University of Toulouse, F-31055 Toulouse, France

Abstract: In this paper, we consider a cyclic job shop problem where a subset of tasks have varying processing times. The minimum processing times and maximum processing times of these tasks are known. We propose a branch and bound method that finds the schedule which minimizes the mean cycle time with respect to variations. We show that the evaluation of a schedule can be considered as a volume calculus of some polytopes. Indeed, for each schedule we can associate a set of polytopes whose volumes provide information on the variation effect on the considered schedule.

© 2017, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.

Keywords: Cyclic scheduling, branch and bound method.

1. INTRODUCTION

Classical scheduling deals with a set of tasks that have to be executed once and optimize a functional object such as makespan, tardiness or maximum tardiness etc. In contrast, cyclic scheduling perform a set of generic tasks that have to be executed infinitely.

Several applications of cyclic scheduling can be found in the literature, e.g. in robotics industry (Dawande et al. (2007); Kats and Levner (2002)) in manufacturing system (Cavory et al. (2005); Chen et al. (1998)), in parallel computing and computer pipelining (Sucha et al. (2004); Ayala et al. (2013); Govindarajan et al. (1996)). It has been studied from multiple points of view. Depending on target application, different mathematical models exist, based on graph theory, mixed linear programming, Petri nets or (max,+) algebra. For more details, an overview about cyclic scheduling and different approaches can be found in Hanen and Munier (1995) and Brucker and Kampsheyer (2008).

This study concerns the Cyclic Job Shop Problem (CJSP). Roundy (1992) investigates a cyclic job shop problem with identical parts. He shows that the problem is NP-hard and proposes a branch and bound to solve the problem. Lee and Posner (1997) study a version of cyclic job shop problem where the execution order of the operations on each machine is given, they show that this problem can be solved in polynomial time. Hanen (1994) and Fink et al. (2012) study a general cyclic job shop problem and propose a branch and bound to solve the problem. A review on the complexity of cyclic scheduling problems can be found in Levner et al. (2010). Different extensions of cyclic job shop problem exist, e.g cyclic job shop with blocking, with limited buffer, no-wait, with transportation, with preemption, etc. Despite the different studies on deterministic cyclic scheduling problems, only few works consider uncertainties. For example Zhang and Graves (1997) study a cyclic scheduling problem with machine breakdowns, Karabati and Tan (1998) study a cyclic scheduling problem with stochastic processing times and Che et al. (2015) study the robust version of cyclic hoist scheduling problem.

More precisely, in this paper, we consider a cyclic job shop problem where a subset of tasks have varying processing times. The minimum processing times and maximum processing times of these tasks are known. The objective considered is to find the schedule which minimizes the mean cycle time with respect to the variations. The main originality of the paper is the way the evaluation of a schedule is performed. We show that this evaluation can be done by computing the volume of some polytopes. More precisely, for each schedule we can associate a set of polytopes whose volumes provide information on the effect of processing times variations on the considered schedule. Then, we derive a branch and bound method to choose the best schedule in the sense of the smallest mean cycle time with respect to the variability of the processing times.

2. CYCLIC SCHEDULING PROBLEMS

2.1 General Basic Cyclic Scheduling Problem (GBCSP)

The General Basic Cyclic Scheduling Problem (GBCSP) is characterized by a set of n generic operations \( \mathcal{T} = \{1, \ldots, n\} \). Each operation \( i \in \mathcal{T} \) has a processing time \( p_i \) and must be repeated infinitely often. The \( k^{th} \) occurrence of the generic operation \( i \) is denoted by \( < i, k \).

A schedule is an assignment of starting time \( t(i,k) \) for each occurrence \( < i, k \) of tasks \( i \in \mathcal{T} \). A schedule is called periodic with cycle time \( \alpha \) if it satisfies

\[
t(i,k) = t(i,0) + \alpha k, \quad \forall i \in \mathcal{T}, \; \forall k \geq 1.
\]

The operations are subjected to a set of generic precedence constraints (uniform constraints). Each of these constraints is represented by a triple \((i,j,H)\) and given by

\[
\begin{align*}
\text{precedence constraints:} & \quad t(i,k) + p_i \leq t(j,k) \\
& \quad t(i,k) + p_i - \delta_i \leq t(j,k)
\end{align*}
\]
\[ t(i, k) + p_i \leq t(j, k + H_{ij}), \quad \forall i \in T, \forall k \geq 1. \]  
(2)

where \( H_{ij} \) is an integer that represents the depth of recurrence, usually referred to as height.

The objective considered in this paper is to find a schedule that minimizes the cycle time \( \alpha \) while satisfying precedence constraints. Other objectives function can be considered, such as work in progress minimization or cycle time and work in progress minimization.

A directed graph \( G = (V,E) \) can be associated with a GBCSP such that a node (resp. an arc) of \( G \) corresponds to a task (resp. constraints) in the GBCSP. Each arc \( (i,j) \) of \( G \) is equipped with two values \( L_{ij} \) and \( H_{ij} \). These arcs are called uniform arcs and are built by considering the precedence constraints. For instance, a precedence constraint between task \( i \) and task \( j \) leads to an arc \( (i, j) \) of \( G \) labeled with \( L_{ij} = p_i \) and \( H_{ij} = 0 \). Two dummy nodes are introduced in the model. More precisely, \( t(s,k) \) represents the start of an occurrence \( k \) and \( t(c,k) \) represents the end of this occurrence. The arc between these two nodes is valued with no processing time and with a positive height denoted \( H^* \). The value \( H^* \) is called Work In Process (WIP) and represents the maximum work in process of the system. Increasing the WIP can influence the cycle time \( \alpha \). We call \( H(c) \) (resp. \( L(c) \)) the height (resp. length) of a circuit \( c \) in graph \( G \) the sum of heights (resp. lengths) of the arcs composing the circuit \( c \).

The following theorem (Hanen (1994)) characterizes a feasible GBCSP.

**Theorem 1.** There exists a feasible schedule if and only if any circuit of \( G \) has a positive height

The minimum cycle time is given by the maximum mean cycle of the graph that is defined by

\[ \alpha = \max_{c \in C} \rho(c) \]

where

\[ \rho(c) = \sum_{(i,j) \in c} L_{ij} - \sum_{(i,j) \in c} H_{ij} \]

and \( C \) is the set of all circuits in \( G \).

The circuit \( c \) that gives the maximum mean cycle is called critical circuit. Several algorithms have been proposed for the computation of critical circuits (see Dasdan and Gupta (1998))

2.2 Cyclic Job Shop Problem (CJSP)

In the present work, we focus on the cyclic job shop problem (CJSP). The difference with the problem defined above is that for CJSP the number of machines is lower than the number of tasks to perform. As a result, the same resource must be shared between the different operations. A CJSP can be considered as a GBCSP endowed with resource constraints.

Each operation \( i \in T \) has a dedicated machine \( M(i) \in M = \{1,\ldots,m\} \) on which its occurrences must be executed. Operations are grouped on a set of jobs \( \mathcal{J} \), where a job \( j \) represents a sequence of elementary operations that must be executed in order. To avoid overlapping between the tasks executed on the same machine, for each pair of operations \( i \) and \( j \) where \( M(i) = M(j) \), the following disjunctive constraint holds

\[ \forall i, j \text{ s.t. } M(i) = M(j), \forall k, l \in \mathbb{N} : t(i,k) \leq t(j,l) \Rightarrow t(i,k) + p_i \leq t(j,l) \]  
(3)

In summary, a cyclic job shop problem is defined by

- a set \( T \) of elementary tasks,
- a set \( M \) of machines,
- for each task \( i \in T \), a processing time \( p_i \) and a machine \( M(i) \in M \) on which the task has to be performed,
- a set \( U \) of uniform constraints,
- a set \( D \) of disjunctive constraints that occur when two tasks are mapped on the same machine,
- a set \( J \) of jobs corresponding to a production sequence of elementary tasks. More precisely, a job \( J_1 \) defines a sequence \( J_1 = t_{1,1} \ldots t_{1,k} \) to be executed in that order.

**Example 2.** We consider an instance of cyclic job shop problem consisting of three jobs. The jobs 1 and 3 have two generic operations and job 2 has four generic operations. Table I summarizes the data of this instance.

As in GBCSP, a directed graph \( G = (V,E) \) can be associated with a CJSP. The uniform arcs are the same as in GBCSP. Additionally, a disjunctive pair of arcs \( (i,j) \) and \( (j,i) \) occurs when the task \( i \) and the task \( j \) are mapped on the same machine. These arcs are labeled respectively with \( L_{ij} = p_i \) and \( H_{ij} = K_{ij} \), and \( L_{ji} = p_j \) and \( H_{ji} = K_{ji} \) where \( K_{ij} \) is occurrence shift variable to determine that satisfy \( K_{ij} + K_{ji} = 1 \) (see Hanen (1994) for further details).

**Example 3.** Fig. 2 represents the graph associated to Example 2. For the sake of clarity, we represent the precedence constraints and only two disjunctive arcs concerning the tasks 1 and 4, but many disjunctive arcs exist between each pair of tasks mapped on the same machine.

**Fink et al. (2012)** have proposed the following bounds on occurrence shift variables \( K_{ij} \):

\[ K_{ij}^- = 1 - \min(H(\mu)| \mu \text{ from } j \text{ to } i \text{ in } G). \]

(4)

Since \( K_{ij} + K_{ji} = 1 \), for each occurrence shifts \( K_{ij} \), the following interval can be derived:

\[ K_{ij}^- \leq K_{ij} \leq 1 - K_{ij}^- \]

(5)
is known, and Toulouse, France, July 9-14, 2017
Proceedings of the 20th IFAC World Congress

For our study, we consider that the durations (Hanen (1994)) for cycle time minimization.

Let us consider the graph $G$ in Fig. 1. The

Example 4. Let us consider the graph $G$ in Fig. 1. The

\begin{align}
\begin{cases}
p_{i} = p_{i} + \delta_{i}z_{i}, & 0 \leq z_{i} \leq 1, & \forall i \in S \\
\end{cases}
\end{align}

According to the variation, the critical circuit may change. Let us consider a task $i$ such that $p_{i} \in [p_{i}, \overline{p}_{i}]$ and $C$ is a circuit that involves the task $i$. The mean cycle of $C$ is then a function of $z_{i}$ and is given by

\begin{align}
\alpha_{C}(z_{i}) = \frac{\sum_{(i,j) \in C} L_{ij}}{\sum_{(i,j) \in C} H_{ij}} + \frac{\delta_{i}z_{i}}{\sum_{(i,j) \in C} H_{ij}}.
\end{align}

Only the circuits containing the task $i$ are affected by the variation of the task $i$. Even if the critical circuit does not contain the task $i$, if the variation is big enough, the circuit $C$ can have a bigger mean value than the critical circuit and thus become the new critical circuit.

Note that the above reasoning still holds if more than one task is time varying.

Therefore, the cycle time of a schedule $\alpha(z)$, where $z$ is a vector of size $|S|$, is a function of the variations ($\alpha(\cdot)$ is

\[V(S) = \int_{0}^{z} \alpha(z) dz.\]
Proceedings of the 20th IFAC World Congress

Let us consider the graph $G$ in Fig. 1. Example 4.

$$|S|$$

vector of size $c$.

The integral calculus is computed with the Vinci library Bueler et al. (2000). This library implements several algorithms. The selected program needs just an hyperplane representation of the polytope or its convex hull definition.

Since we seek to determine a solution that has the smallest cycle time mean, this measure corresponds to our objective as

$$\frac{1}{\prod_{i \in S \setminus \emptyset} f} \times \int_0^1 q(z) dz$$

represents the mean value of cycle time over the variation of $p_i$.

The function $\alpha(\cdot)$ is built from an exact enumeration of circuits that involve the task $i$. For the enumeration of these circuits the Tarjan’s algorithm (Tarjan (1973)) or Johnson’s algorithm (Johnson (1975)) are used.

Between two solutions, the best one is the one that has the smallest volume because its cycle time is smaller in average. For this reason, our objective is to find a schedule that minimizes the associated volume.

Example 6. Let us consider the problem described in Table 1.

The optimal solution $\mathcal{S}$ of the problem without variations is given by:

$K_{12} = K_{26} = K_{67} = K_{45} = K_{84} = K_{85} = 0$, $K_{61} = K_{71} = K_{72} = K_{43} = K_{53} = K_{33} = 2$  (the rest of occurrence shift values can be deduced by formul\(a) (4) and (5) ) and the associated volume is $V(\mathcal{S}) = 4242$.

Now, if we consider the problem with maximum variations, the optimal solution $\mathcal{S}$ is given by:

$K_{21} = K_{17} = K_{76} = K_{43} = K_{35} = K_{45} = K_{58} = 0$, $K_{63} = K_{62} = K_{72} = K_{83} = K_{84} = 2$ and the associated volume is $V(\mathcal{S}) = 4278$.

Finally, the optimal solution $S_{opt}$ of the problem is given by:

$K_{45} = K_{12} = K_{26} = K_{67} = K_{16} = K_{35} = K_{44} = K_{38} = K_{85} = K_{84} = 0$, $K_{71} = K_{72} = 2$ and the associated volume is $v(S_{opt}) = 4071$.

The three schedules are represented respectively in Fig. 3, Fig. 4 and Fig. 5.

5. SOLUTION APPROACH

In this section, we describe a branch and bound procedure that finds the schedule $S_{opt}$ minimizing the associated volume. A pseudo-code of the procedure is presented in Fig. 6.

We consider that the durations of a subset of tasks $\mathcal{S} \subseteq \mathcal{T}$ are varying and the deviation of each task in $\mathcal{S}$ is assumed to be known. In order to initialize the search tree, a node $R$ (a root) is created by considering the problem without the disjunctive constraints (that is an obvious relaxation of the problem) but only the uniform constraints. This problem is equivalent to GBCSP.

Determining bounds on the volume value of optimal solutions allow us to prune branches of the search tree and thus reduce the search area. For this purpose, we consider a problem CJSP$_{max}$ (resp.CJSP$_{min}$) where all processing times are fixed to upper (resp. lower) bounds and let $\mathcal{S}$ (resp. $\mathcal{S}$), the associate optimal solution. Both $\mathcal{S}$ and $\mathcal{S}$ are feasible for the time varying cyclic job shop, therefore their associated volume value $V(\mathcal{S})$ and $V(\mathcal{S})$ represent upper bounds on the optimal volume value $V(S_{opt})$. Note that there is no dominance between these two bounds. The minimum of $V(\mathcal{S})$ and $V(\mathcal{S})$ is set as the upper bound of $V(S_{opt})$.

The same branch and bound scheme as in Fink et al. (2012) is used for solving the problem. More precisely, for each shift event $K_{ij}$, corresponding to a disjunctive arc in graph $G$, an upper bound $K_{ij}^+$ and lower bound $K_{ij}^-$ can be deduced (Section 2.2).

Branching is performed by fixing a value of occurrence shift $K_{ij}$ on interval $\mathcal{I}_{ij} = [K_{ij}^-, 1 - K_{ij}^+]$, i.e. by adding a disjunctive arc with a fixed shift event $K_{ij} \in \mathcal{I}_{ij}$ to a graph $G$. A new child node is created for each integer in the interval $\mathcal{I}_{ij}$. Then, each child node is evaluated. This step is achieved using the measure introduced in the previous section, i.e. by computing the volume of a polytope formed by a set of linear constraints. The Vinci library is used for this purpose. Note that each volume of non-root and non-leaf node represents a lower bound on the minimal volume.

Concerning the branching rule, we choose to select the undetermined shift event $K_{ij}$ such that $K_{ij}^+ + K_{ij}^-$ is maximal. This branching rule induces a minimal number of child nodes, which can lead to the smallest search tree.

When all occurrence shifts are fixed (i.e. we reach a leaf node) a complete schedule is obtained. This schedule is a feasible and the associated volume represents an upper bound. If this volume improve the best upper bound, we update it.

The best solution of the procedure corresponds to the schedule with lowest volume of the polyhedron generated. As we have noticed in the previous section, this schedule is also the schedule with the minimal mean value.

6. CONCLUSION AND FUTURE WORKS

In this paper, we have proposed a branch and bound procedure that finds a schedule minimizing the mean cycle time with respect to processing times variations. This
Algorithm 1

Data: $G = (V, E)$
Result: A schedule $S_{opt}$ with a minimum volume

begin
\[
\begin{align*}
&|\text{Initialization} \\
&UB \leftarrow \min(V(S), V(S)) \\
&\text{Initialize the first node R} \\
&\text{Stack.push.back(R);} \\
&\text{while (Stack ≠ ∅) do} \\
&\quad S \leftarrow \text{Stack.top()} \\
&\quad \text{if } V(S) < UB \text{ then} \\
&\quad \quad \text{if } S \text{ is a complete solution then} \\
&\quad \quad \quad UB \leftarrow V(S) \\
&\quad \quad \text{else} \\
&\quad \quad \quad S(\text{selected } K_{ij}) \leftarrow \text{branchingRule}(S) \\
&\quad \quad \ed{evaluatenode(N)} \\
&\quad \text{Stack.push.back(nodeSelection(N))} \\
\end{align*}
\]

end

Fig. 6. branch and bound procedure for cyclic job shop with varying processing times

procedure uses a new a performance measure of schedule based on volume calculus of polytope.

Further works will concern the numerical experimentation of the branch and bound method and the consideration of some extensions of the problem. For example we can introduce probability for each possible value of processing times and adapt the presented performance measure of schedule. We can also consider other criteria such as, maximizing the number of scenarios were a scenario is an instantiation of all the varying processing times such that the cycle time is below a given level or maximizing the range of value $\delta_k$ around a point of interest such that the cycle time remains unchanged.

REFERENCES


