Cyclic Job Shop Problem with varying processing times

Idir Hamaz, Laurent Houssin, Sonia Cafieri

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 50-1 (2017) 5012–5016

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.

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 an objective function 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 Kampmeyer (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 $\mathcal{T} = \{1, ..., n\}$. 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 α if it satisfies

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

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

10.1016/j.ifacol.2017.08.908

$$t(i,k) + p_i \leqslant t(j, k + H_{ij}), \quad \forall i \in \mathcal{T},\ \forall k \geq 1. \qquad (2)$$

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 α 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 $t(e, k)$ represents the end of this occurrence. The arc between these two nodes is valuated 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 α. 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

$$\alpha = \max_{c \in \mathcal{C}} \rho(c)$$

where

$$\rho(c) = \frac{\sum_{(i,j) \in c} L_{ij}}{\sum_{(i,j) \in c} H_{ij}}$$

and $\mathcal{C}$ 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 $i \in \mathcal{T}$ has a dedicated machine $M_{(i)} \in \mathcal{M} = \{1, ..., m\}$ 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

Table 1. Data for instance given in Example 1

Job	Task	Processing time	Machine
1	1	12	2
	2	6	2
3	3	9	1
	4	13	2
	5	16	2
	6	15	1
2	7	14	1
	8	11	2

$$\forall\, i, j \ s.t.\ M_{(i)} = M_{(j)},\ \forall k, l \in \mathbb{N} :$$
$$t(i,k) \leq t(j,l) \Rightarrow t(i,k) + p_i \leq t(j,l) \qquad (3)$$

In summary, a cyclic job shop problem is defined by

- a set $\mathcal{T}$ of elementary tasks,
- a set $\mathcal{M}$ of machines,
- for each task $i \in \mathcal{T}$, a processing time p_i and a machine $M_{(i)} \in \mathcal{M}$ on which the task has to be performed,
- a set $\mathcal{U}$ of uniform constraints,
- a set $\mathcal{D}$ of disjunctive constraints that occur when two tasks are mapped on the same machine,
- a set $\mathcal{J}$ 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}) :

$$K_{ij}^{-} = 1 - min\{H(\mu) \,|\, \mu \ from\ j\ to\ i\ in\ G\}. \qquad (4)$$

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

$$K_{ij}^{-} \leq K_{ij} \leq 1 - K_{ij}^{-}. \qquad (5)$$

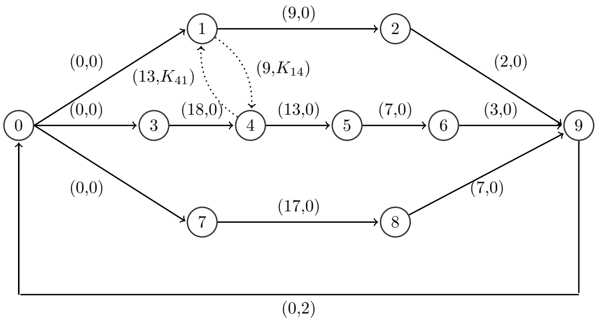

Fig. 1. Asociated graph to instance in Example 2.

Example 4. Let us consider the graph G in Fig. 1 . The height of the circuit $\mathcal{C} = \{0, 3, 4, 1, 2, 9, 0\}$, is $H(\mathcal{C}) = 0 + 0 + K_{41}$+0+0+1. According to Theorem 1, $H(\mathcal{C}) = 1 + K_{24} \geq 1$. Thus, we deduce $K_{24} \geq 0$. As we know that $K_{ij} + K_{ji} = 1$, we deduce that $0 \leq K_{ij} \leq 1$.

A schedule is an assignment of all the occurrence shifts, ie. fix a precedence relations between operations mapped to the same machine. Note that once the occurrence shifts are determined, the minimum cycle time can be obtained by computing the critical circuit of the associated graph. Since the problem is cyclic, a schedule is defined only with $t_i = t(i, 0)$ for all $i \in \mathcal{T}$ and α.

Previous studies have shown that the problem is NP-Hard (Hanen (1994)) for cycle time minimization.

3. CJSP WITH UNCERTAIN PROCESSING TIMES

Uncertainties on processing time can have a negative impact on activity duration. Indeed, a good solution to the problem without variation can easily become the worst solution when uncertainties occurs.
For our study, we consider that the durations p_i of a subset of tasks $\mathcal{S} \subseteq \mathcal{T}$ are affected by uncertainties. We model the uncertain parameter $p_i, i \in \mathcal{S}$, as a value belonging to the interval $[\underline{p_i}, \overline{p_i}]$. The bound $\underline{p_i}$ is known, and $\overline{p_i} = \underline{p_i} + \delta_i$ where δ_i is the maximum variation. Equivalently

$$p_i = \underline{p_i} + \delta_i z_i,\ 0 \leq z_i \leq 1,\ \forall i \in \mathcal{S} \tag{6}$$

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

$$\alpha_{\mathcal{C}}(z_i) = \frac{\sum_{(i,j)\in\mathcal{C}} L_{ij}}{\sum_{(i,j)\in\mathcal{C}} H_{ij}} + \frac{\delta_i z_i}{\sum_{(i,j)\in\mathcal{C}} H_{ij}.} \tag{7}$$

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 $\mathcal{C}$ 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 $|\mathcal{S}|$, is a function of the variations ($\alpha(\cdot)$ is piecewise affine). An illustration of a function $\alpha(\cdot)$ when $|\mathcal{S}| = 1$ is drawn in Fig. 2. We can distinguish that three circuits can be critical according to the value of p_i.

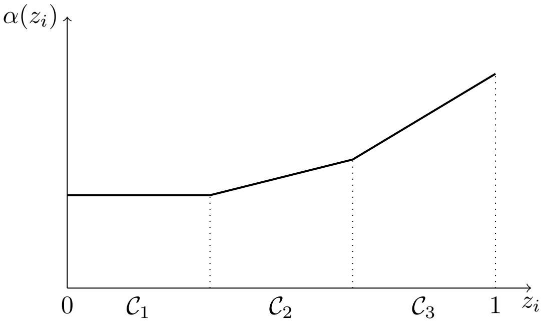

Fig. 2. The cycle time is a function of the variation of the task i.

Example 5. We consider the instance described in Table 1, and we suppose that the tasks 5 and 7 are varying. The maximum variation values are respectively, 15 and 6. In order to show the impact on the critical circuit in case of the cyclic job shop problem, we consider two cases. In the first case, we consider that the variation is zero for both the varying tasks. The critical circuit is $C_2 = (0, 7, 2, 3, 6, 9, 0)$ and the cycle time $\alpha = \frac{40}{1} = 40$. As a second case, we consider that the variations take their maximum values. More precisely, the task 2 will take a processing time 27 and tasks 5 will take 14. In this case, the critical circuit is $\mathcal{C}_1 = (1, 8, 4, 5, 1)$ and the cycle time $\alpha = \frac{51}{1} = 51$.

4. A PERFORMANCE MEASURE OF SCHEDULE:VOLUME COMPUTING

In cyclic scheduling literature (Hanen and Munier (1995)), two classical methods for performance evaluation of deterministic cyclic job shop problem can be found. The first one is based on graph theory. In this framework, the cycle time computation becomes the search of maximum circuit ratio in graph. The second approach considers the cyclic job shop as $(max, +)$-linear system. In this case, the spectrum of evolution matrix of the system gives the cycle time. Both approaches have similar complexity and depend on the procedure to compute maximum circuit ratio of the graph or the eigenvalue of the evolution matrix. We can mention the Howard algorithm (Cochet-Terrasson et al. (1998)). The complexity of this algorithm is not proved but it shows excellent practical performance.

As these approaches measure the performance of a schedule only for fixed processing times, we aim to propose a new performance measure of schedule, that can provide an information about a schedule with respect to the variations. The idea is to look through the evolution of the cycle time with respect to processing times variations and measure the mean of cycle time. For this purpose, we can consider the volume of the polytope representing the cycle time evolution with respect to variations. This volume is given by

$$V(S) = \int_{\mathbb{0}}^{\mathbb{1}} \alpha(z)dz.$$

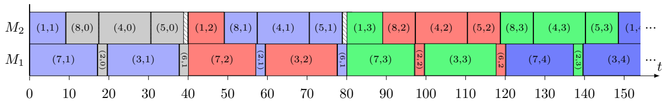

Fig. 3. Optimal schedule $\underline{S}$ for Example 2 without variations.

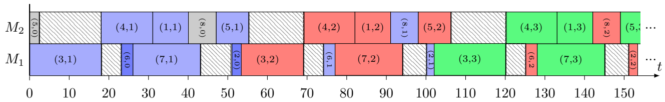

Fig. 4. Optimal schedule $\overline{S}$ (Example 2) with maximal variations.

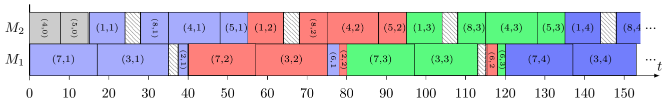

Fig. 5. Optimal schedule S_{opt} for Example 2.

Where $\mathbb{0}$ is vector of zeros and $\mathbb{1}$ vector of ones. The size of both vectors is $|\mathcal{S}|$.

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 $\frac{1}{\Pi_{i \in \mathcal{S}} \delta_i} \times \int_{\mathbb{0}}^{\mathbb{1}} \alpha(\underline{z}) d\underline{z}$ 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 $\underline{S}$ of the problem without variations is given by :
$K_{12} = K_{26} = K_{67} = K_{45} = K_{84} = K_{85} = 0$, $K_{61} = K_{71} = K_{72} = K_{43} = K_{53} = K_{83} = 2$ (the rest of occurrence shift values can be deduced by formula (4) and (5)) and the associated volume is $V(\underline{S}) = 4242$.

Now, if we consider the problem with maximum variations, the optimal solution $\overline{S}$ is given by :
$K_{21} = K_{17} = K_{76} = K_{43} = K_{35} = K_{45} = K_{58} = 0$, $K_{61} = K_{62} = K_{72} = K_{83} = K_{84} = 2$ and the associated volume is $V(\overline{S}) = 4278$.
Finally, the optimal solution S_{opt} of the problem is given by :
$K_{45} = K_{12} = K_{26} = K_{67} = K_{16} = K_{35} = K_{34} = K_{38} = K_{85} = K_{84} = 0$, $K_{71} = K_{72} = 2$ 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 $\overline{S}$ (resp. $\underline{S}$), the associate optimal solution. Both $\overline{S}$ and $\underline{S}$ are feasible for the time varying cyclic job shop, therefore their associated volume values $V(\overline{S})$ and $V(\underline{S})$ represent upper bounds on the optimal volume value $V(S_{opt})$. Note that there is no dominance between these two bounds. The minimum of $V(\overline{S})$ and $V(\underline{S})$ 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 $K^-_{ij} + K^-_{ji}$ 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 feasable 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
    \\ Initialization
    $UB \longleftarrow \min(V(\overline{S}), V(\underline{S}))$
    Initialize the first node R
    Stack.push_back(R);
    while ($Stack \neq \varnothing$) do
        $S \longleftarrow$ Stack.top()
        if $V(S) < UB$ then
            if $S$ is a complete solution then
                $UB \longleftarrow V(S)$
            else
                $S$(selected $K_{ij}$) $\longleftarrow$ branchingRule($S$)
                $N \longleftarrow$branch($S$)
                evaluateNode($N$)
                Stack.push_back(nodeSelection($N$))
```

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.

Furthers works will concerns 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 δ_i around a point of interest such that the cycle time remains unchanged.

REFERENCES

Ayala, M., Benabid, A., Artigues, C., and Hanen, C. (2013). The resource-constrained modulo scheduling problem: an experimental study. *Computational Optimization and Applications*, 54(3), 645–673.

Brucker, P. and Kampmeyer, T. (2008). A general model for cyclic machine scheduling problems. *Discrete Applied Mathematics*, 156(13), 2561 – 2572.

Bueler, B., Enge, A., and Fukuda, K. (2000). Exact volume computation for polytopes: A practical study. *Polytopes — Combinatorics and Computation*, 29.

Cavory, G., Dupas, R., and Goncalves, G. (2005). A genetic approach to solving the problem of cyclic job shop scheduling with linear constraints. *European Journal of Operational Research*, 161(1), 73 – 85.

Che, A., Feng, J., Chen, H., and Chu, C. (2015). Robust optimization for the cyclic hoist scheduling problem. *European Journal of Operational Research*, 240(3), 627–636.

Chen, H., Chu, C., and Proth, J. (1998). Cyclic scheduling of a hoist with time window constraints. *IEEE Trans. Robotics and Automation*, 14(1), 144–152.

Cochet-Terrasson, J., Cohen, G., Gaubert, S., Gettrick, M.M., and Quadrat, J.P. (1998). Numerical computation of spectral elements in max-plus algebra. In *Proceedings of the IFAC Conference on System Structure and Control*. Nantes.

Dasdan, A. and Gupta, R.K. (1998). Faster maximum and minimum mean cycle algorithms for system-performance analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(10), 889–899.

Dawande, M.W., Geismar, H.N., Sethi, S.P., and Sriskandarajah, C. (2007). *Throughput optimization in robotic cells*, volume 101. Springer Science & Business Media.

Fink, M., Rahhou, T.B., and Houssin, L. (2012). A new procedure for the cyclic job shop problem. In *Proceedings of INCOM 2012*, volume 14. Bucharest, Romania.

Govindarajan, R., Altman, E.R., and Gao, G.R. (1996). A framework for resource-constrained rate-optimal software pipelining. *IEEE Transactions on Parallel and distributed systems*, 7(11), 1133–1149.

Hanen, C. (1994). Study of a np-hard cyclic scheduling problem: The recurrent job-shop. *European Journal of Operational Research*, 72(1), 82 – 101.

Hanen, C. and Munier, A. (1995). *Scheduling Theory and Its Applications*, chapter Cyclic Scheduling on Parallel Processors: An Overview. Wiley.

Johnson, D.B. (1975). Finding all the elementary circuits of a directed graph. *SIAM Journal on Computing*, 4(1), 77–84.

Karabati, S. and Tan, B. (1998). Stochastic cyclic scheduling problem in synchronous assembly and production lines. *Journal of the Operational Research Society*, 49(11), 1173–1187.

Kats, V. and Levner, E. (2002). Cyclic scheduling in a robotic production line. *Journal of Scheduling*, 5(1), 23–41.

Lee, T.E. and Posner, M.E. (1997). Performance measures and schedules in periodic job shops. *Operations Research*, 45(1), 72–91.

Levner, E., Kats, V., de Pablo, D.A.L., and Cheng, T.E. (2010). Complexity of cyclic scheduling problems: A state-of-the-art survey. *Computers & Industrial Engineering*, 59(2), 352–361.

Roundy, R. (1992). Cyclic schedules for job shops with identical jobs. *Mathematics of Operations Research*, 17(4), 842–865.

Sucha, P., Pohl, Z., and Hanzálek, Z. (2004). Scheduling of iterative algorithms on fpga with pipelined arithmetic unit. In *Real-Time and Embedded Technology and Applications Symposium, 2004. Proceedings. RTAS 2004. 10th IEEE*, 404–412. IEEE.

Tarjan, R. (1973). Enumeration of the elementary circuits of a directed graph. *SIAM Journal on Computing*, 2(3), 211–216.

Zhang, H. and Graves, S.C. (1997). Cyclic scheduling in a stochastic environment. *Operations Research*, 45(6), 894–903.