A Branch-and-Bound Procedure for the Robust Cyclic Job Shop Problem

Idir Hamaz, Laurent Houssin, Sonia Cafieri

A Branch-and-Bound procedure for the robust cyclic job shop problem

Idir Hamaz[1], Laurent Houssin[1] and Sonia Cafieri[2]

[1] LAAS-CNRS, Universite de Toulouse, CNRS, UPS, Toulouse, France
{ihamaz, lhoussin}@laas.fr
[2] ENAC, Universite de Toulouse, F-31055 Toulouse, France
sonia.cafieri@enac.fr

Abstract. This paper deals with the cyclic job shop problem where the task durations are uncertain and belong to a polyhedral uncertainty set. We formulate the cyclic job shop problem as a two-stage robust optimization model. The cycle time and the execution order of tasks executed on the same machines correspond to the *here-and-now* decisions and have to be decided before the realization of the uncertainty. The starting times of tasks corresponding to the *wait-and-see* decisions are delayed and can be adjusted after the uncertain parameters are known. In the last decades, different solution approaches have been developed for two-stage robust optimization problems. Among them, the use of affine policies, column generation algorithms, row and row-and-column generation algorithms. In this paper, we propose a Branch-and-Bound algorithm to tackle the robust cyclic job shop problem with cycle time minimization. The algorithm uses, at each node of the search tree, a robust version of the Howard's algorithm to derive a lower bound on the optimal cycle time. We also develop a heuristic method that permits to compute an initial upper bound for the cycle time. Finally, encouraging preliminary results on numerical experiments performed on randomly generated instances are presented.

Keywords: Cyclic job shop problem, robust optimization, Branch-and-Bound algorithm.

1 Introduction

Most models for scheduling problems assume deterministic parameters. In contrast, real world scheduling problems are often subject to many sources of uncertainty. For instance, activity durations can decrease or increase, machines can break down, new activities can be incorporated, *etc.* In this paper, we focus on scheduling problems that are cyclic and where activity durations are affected by uncertainty. Indeed, the best solution for a deterministic problem can quickly become the worst one in the presence of uncertainties.

In this work, we focus on the *Cyclic Job Shop Problem* (CJSP) where processing times are affected by uncertainty. Several studies have been conducted on the

CJSP in its deterministic setting. The CJSP with identical jobs is studied in [1] and the author shows that the problem is $\mathcal{NP}$-hard and proposes a Branch-and-Bound algorithm to solve the problem. The more general CJSP is investigated in [2], where the author proposes a mixed linear integer programming formulation and presents a Branch-and-Bound procedure to tackle the problem. A general framework for modeling and solving cyclic scheduling problems is presented in [3]. The authors present different models for cyclic versions of CJSP. However, a few works consider cyclic scheduling problems under uncertainty. The cyclic hoist scheduling problem with processing time window constraints where the hoist transportation times are uncertain has been investigated by Che et al. [4]. The authors define a robustness measure for cyclic hoist schedule and present a bi-objective mixed integer linear programming model to optimize both the cycle time and the robustness.

Two general frameworks have been introduced to tackle optimization problems under uncertainty. These frameworks are Stochastic Programming (SP) and Robust Optimization (RO). The main difference between the two approaches is that the Stochastic Programming requires the probability description of the uncertain parameters while RO does not. In this paper, we focus on the RO paradigm. More precisely, we model the robust CJSP as a two-stage RO problem. The cycle time and the execution order of tasks on the machines corresponding to the *here-and-now* decisions have to be decided before the realization of the uncertainty, while the starting times of tasks corresponding to the *wait-and see* decisions are delayed and can be adjusted after the uncertain parameters are known. In recent years there has been a growing interest in the two-stage RO and in the multi-stage RO in general. The two-stage RO is introduced in [5], referred to as *adjustable optimization*, to address the over-conservatism of single stage RO models. Unfortunately, the two-stage RO problems tend to be intractable [5]. In order to deal with this issue, the use of affine policies ([5]) and decomposition algorithms ([6], [7],[8]) have been proposed.

This paper deals with the CJSP where the task durations are uncertain and belong to a polyhedral uncertainty set. The objective is to find a minimum cycle time and an execution order of tasks executed on the same machines such that a schedule exists for each possible scenario in the uncertainty set. To tackle the problem we design a Branch-and-Bound algorithm. More precisely, at each node of the search tree, we solve a robust Basic Cyclic Scheduling Problem (BCSP), which corresponds to the CJSP without resource constraints, using a robust version of Howard's algorithm to get a lower bound. We also propose a heuristic algorithm to find an initial upper bound on the cycle time. Finally, we provide results on numerical experiments performed on randomly generated instances.

This paper is structured as follows. In Section 2, we present both the Basic Cyclic Scheduling Problem and the Cyclic Job Shop Problem in their deterministic case and introduce the polyhedral uncertainty set considered in this study. Section 3 describes a Branch-and-Bound ($B\&B$) procedure to solve the robust CJSP. Numerical experiments performed on randomly generated instances are reported and discussed in Section 4. Finally, some concluding remarks and perspectives are drawn in Section 5.

2 The cyclic scheduling problems

In this section, we first introduce the Basic Cyclic Scheduling Problem which corresponds to the CJSP without resource constraints. This problem will represent a basis for the Branch-and-Bound method designed for the robust CJSP solving. Next, we present the CJSP in its deterministic case. Finally, we present the uncertainty set that we consider in this paper and we formulate the CJSP with uncertain processing times as a two-stage robust optimization problem.

2.1 Basic Cyclic Scheduling Problem (BCSP)

Let $\mathcal{T} = \{1, ..., n\}$ be a set of n generic operations. Each operation $i \in \mathcal{T}$ has a processing time p_i and must be performed infinitely often. We denote by $< i, k >$ The k^{th} occurrence of the operation i and by $t(i, k)$ the starting time of $< i, k >$.

The operations are linked by a set $\mathcal{P}$ of *precedence constraints* (uniform constraints) given by

$$t(i, k) + p_i \leqslant t(j, k + H_{ij}), \quad \forall (i, j) \in \mathcal{P},\ \forall k \geq 1, \tag{1}$$

where i and j are two generic tasks and H_{ij} is an integer representing the depth of recurrence, usually referred to as *height*.

Furthermore, two successive occurrences of the same task i are not allowed to overlap. This constraint corresponds to the non-reentrance constraint and can be modeled as a uniform constraint with a height $H_{ii} = 1$.

A schedule S is an assignment of starting time $t(i, k)$ for each occurrence $< i, k >$ of tasks $i \in \mathcal{T}$ such that the precedence constraints are met. A schedule S is called *periodic* with cycle time α if it satisfies

$$t(i, k) = t(i, 0) + \alpha k, \quad \forall i \in \mathcal{T},\ \forall k \geq 1. \tag{2}$$

For the sake of simplicity, we denote by t_i the starting time of the occurrence $< i, 0 >$. Since the schedule is periodic, a schedule can be completely defined by the vector of the starting times $(t_i)_{i \in \mathcal{T}}$ and the cycle time α.

The objective of the BCSP is to find a schedule that minimizes the cycle time α while satisfying precedence constraints. Note that other objective functions can be considered, such as work-in-process minimization [2].

A directed graph $G = (\mathcal{T}, \mathcal{P})$, called *uniform graph*, can be associated with a BCSP such that each node $v \in \mathcal{T}$ (resp. arc $(i, j) \in \mathcal{P}$) corresponds to a generic task (resp. uniform constraint) in the BCSP. Each arc $(i, j) \in \mathcal{P}$ is labeled with two values, a *length* $L_{ij} = p_i$ and a *height* H_{ij}.

We denote by $L(c)$ (resp. $H(c)$) the length (resp. height) of a circuit c in graph G, representing the sum of lengths (resp. heights) of the arcs composing the circuit c.

Let us recall the necessary and sufficient condition for the existence of a feasible schedule.

Theorem 1 (Hanen C. [2]). *There exists a feasible schedule if and only if any circuit of G has a positive height.*

A graph that satisfies the condition of Theorem 1 is called consistent. In the following, we assume that the graph G is always consistent. In other words, a feasible schedule always exists.

The minimum cycle time is given by the maximum circuit ratio of the graph G that is defined by

$$\alpha = \max_{c \in \mathcal{C}} \frac{L(c)}{H(c)}$$

where $\mathcal{C}$ is the set of all circuits in G. The circuit c with the maximum circuit ratio is called a *critical circuit*. Thus, the identification of the critical circuit in graph G allows one to compute the minimum cycle time.

Many algorithms for the computation of the cycle time and the critical circuit can be found in the literature. A binary search algorithm with time complexity $\mathcal{O}(nm\left(log(n) + log(\max_{(i,j)\in E}(L_{ij}, H_{ij}))\right))$ has been proposed in[9]. Experimental study about maximum circuit ratio algorithms has been presented in [10]. This study shows that the Howard's algorithm is the most efficient among the tested algorithms.

Once the optimal cycle time α is determined by one of the algorithms cited above, the optimal periodic schedule can obtained by computing the longest path in the graph $G = (\mathcal{T}, \mathcal{P})$ where each arc $(i,j) \in \mathcal{P}$ is weighted by $p_i - \alpha H_{ij}$.

The BCSP can also be solved by using the following linear program:

$$\min \quad \alpha \tag{3}$$
$$s.t.\ t_j - t_i + \alpha H_{ij} \geq p_i \quad \forall (i,j) \in \mathcal{P} \tag{4}$$

where t_i represents $t(i,0)$, i.e., the starting time of the first occurrence of the task i. Note that the precedence constraints (4) are obtained by replacing in (1) the expression of $t(i,k)$ given in (2).

2.2 Cyclic Job Shop Problem (CJSP)

In the present work, we focus on the cyclic job shop problem (CJSP). Contrary to the BCSP, in the CJSP, the number of machines is lower than the number of tasks. As a result, an execution order of the operations executed on the same machine have to be determined.

Each occurrence of an operation $i \in \mathcal{T} = \{1, ..., n\}$ has to be executed, without preemption, on the machine $M_{(i)} \in \mathcal{M} = \{1, ..., m\}$. Operations are grouped on a set of jobs $\mathcal{J}$, where a job $j \in \mathcal{J}$ represents a sequence of generic operations that must be executed in a given 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

$$\forall\, i,j \ s.t.\ M_{(i)} = M_{(j)},\ \forall k,l \in \mathbb{N} : t(i,k) \leq t(j,l) \Rightarrow t(i,k) + p_i \leq t(j,l) \tag{5}$$

To summarize, the CJSP is defined by

- a set $\mathcal{T} = \{1, ..., n\}$ of n generic tasks,
- a set $\mathcal{M} = \{1, ..., m\}$ of m machines,
- each task $i \in \mathcal{T}$ has a processing time p_i and has to be executed on the machine $M_{(i)} \in \mathcal{M}$,
- a set $\mathcal{P}$ of precedence 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 sequence of elementary tasks. More precisely, a job J_j defines a sequence $J_j = O_{j,1} \dots O_{j,k}$ of operations that have to be executed in that order.

The CJSP can be represented by directed graph $G = (\mathcal{T}, \mathcal{P} \cup \mathcal{D})$, called *disjunctive graph*. The sequence of operations that belong to the same job are linked by uniform arcs in $\mathcal{P}$ where the heights are equal to 0. Additionally, for each pair of operations i and j executed on the same machine, a disjunctive pair of arcs (i, j) and (j, i) occurs. 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 an occurrence shift variable to determine that satisfy $K_{ij} + K_{ji} = 1$ (see [2] for further details). Note that the K_{ij} variables are integer variables and not binary variables as is the case for the non-cyclic job shop problem. Two dummy nodes s and e representing respectively the start and the end of the execution are added to the graph. An additional arc (e,s) with $L_{es} = 0$ and $H_{ij} = WIP$ is considered. The WIP parameter is an integer, called a work-in-process, and represents the number of occurrences of a job concurrently executed in the system.

A lower bound on each occurrence shift value K_{ij} that makes the graph G consistent can be obtained as follows (see [2],[11]):

$$K^-_{ij} = 1 - min\{H(\mu) \,|\, \mu \textit{ is a path from } j \textit{ to } i \textit{ in } G = (\mathcal{T}, \mathcal{P} \cup \emptyset)\}. \tag{6}$$

Since $K_{ij} + K_{ji} = 1$, one can deduce an upper bound:

$$K^-_{ij} \leq K_{ij} \leq 1 - K^-_{ji}. \tag{7}$$

The objective of the problem is to find an assignment of all the occurrence shifts, in other words, determining an order on the execution of operations mapped to the same machine such that the cycle time is minimum. Note that, once the occurrence shifts are determined, the minimum cycle time can be obtained by computing the critical circuit of the associated graph G.

2.3 CJSP problem with uncertain processing times ($\mathcal{U}^\Gamma$-CJSP)

We define the uncertainty set through the *budget of uncertainty* concept introduced in [12]. The processing times $(p_i)_{i \in \mathcal{T}}$ are uncertain and each processing time p_i belongs to the interval $[\bar{p}_i, \bar{p}_i + \hat{p}_i]$, where $\bar{p}_i$ is the nominal value and $\hat{p}_i$ the deviation of the processing time p_i from its nominal value. We associate a binary variable ξ_i to each operation $i \in \mathcal{T}$. The variable ξ_i is equal to 1 if the processing time of the operation i takes its worst-case value, 0 otherwise. For a given budget of uncertainty Γ, that is a positive integer representing the maximum number of tasks allowed to take their worst-case values, the processing time deviations can be modeled trough the following uncertainty set:

$$\mathcal{U}^{\Gamma} = \left\{ (p_i)_{i \in \mathcal{T}} \in \mathbb{R}^n : p_i = \bar{p}_i + \hat{p}_i \xi_i, \, \forall\, i \in \mathcal{T};\, \xi_i \in \{0,1\};\, \sum_{i \in \mathcal{T}} \xi_i \leq \Gamma \right\}.$$

The BCSP problem under the uncertainty set $\mathcal{U}^{\Gamma}$ is studied in [13]. Three exact algorithms are proposed to solve the problem. Two of them use a negative circuit detection algorithm as a subroutine and the last one is a Howard's algorithm adaptation. Results of numerical experiments show that the Howard algorithm adaptation yields efficient results.

The problem we want to solve in this study can be casted as follows:

$$\min \quad \alpha \tag{8}$$

$$s.t.\ \forall\, p \in \mathcal{U}^{\Gamma} : \exists\, t \geq 0 \begin{cases} t_j - t_i + \alpha H_{ij} \geq p_i & \forall\, (i,j) \in \mathcal{P} \\ t_j - t_i + \alpha K_{ij} \geq p_i & \forall\, (i,j) \in \mathcal{D} \end{cases} \tag{9}$$

$$K_{ij} + K_{ji} = 1 \qquad \forall\, (i,j) \in \mathcal{D} \tag{10}$$

$$K^{-}_{ij} \leq K_{ij} \leq 1 - K^{-}_{ji} \qquad \forall\, (i,j) \in \mathcal{D} \tag{11}$$

$$K_{ij} \in \mathbb{Z} \qquad \forall\, (i,j) \in \mathcal{D} \tag{12}$$

$$\alpha \geq 0 \tag{13}$$

In other words, we aim to find a cycle α and occurrence shifts $(K_{ij})_{(i,j)\in\mathcal{D}}$ such that, for each possible value of the processing times $p \in \mathcal{U}^{\Gamma}$, there always exists a feasible vector of starting time $(t_i)_{i\in\mathcal{T}}$.

Note that, once the occurrence shifts are fixed, the problem can be solved as a robust BCSP by using the algorithms described in [13]. The following theorem characterizes the value of the optimal cycle time for $\mathcal{U}^{\Gamma}$-CJSP:

Theorem 2 ([13]). *The optimal cycle time α of the $\mathcal{U}^{\Gamma}$-CJSP is characterized by*

$$\alpha = \max_{c \in \mathcal{C}} \left\{ \frac{\sum\limits_{(i,j)\in c} \bar{L}_{ij}}{\sum\limits_{(i,j)\in c} H_{ij}} + \max_{\xi : \sum_{i\in\mathcal{T}} \xi_i \leq \Gamma} \left\{ \frac{\sum\limits_{(i,j)\in c} \hat{L}_{ij}\xi_i}{\sum\limits_{(i,j)\in c} H_{ij}} \right\} \right\},$$

where $\bar{L}_{ij} = \bar{p}_i$, $\hat{L}_{ij} = \hat{p}_i$ and $\mathcal{C}$ is the set of all circuits in G.

3 Branch-and-Bound method

We develop a Branch-and-Bound algorithm for solving $\mathcal{U}^{\Gamma}$-CJSP. Each node of the Branch-and-Bound corresponds to a subproblem defined by the subgraph

$G_s = (\mathcal{T}, \mathcal{P} \cup \mathcal{D}_s)$, where $\mathcal{D}_s \subseteq \mathcal{D}$ is a subset of occurrence shifts already fixed. The algorithm starts with a root node G_{root} where $\mathcal{D}_{root} = \emptyset$, in other words, no occurrence shifts are fixed. The branching is performed by fixing an undetermined occurrence shift K_{ij} and creates a child node for each possible value of K_{ij} in $[K_{ij}^-, 1 - K_{ji}^-]$. Each of these nodes is evaluated by computing the associated cycle time, such that a schedule exists for each $p \in \mathcal{U}^\Gamma$. This evaluation is made by means of the robust version of Howard's algorithm. Our method explores the search tree in best-first search (BeFS) manner, and, in order to branch, it chooses the node having the smallest lower bound. This search strategy can lead to a good feasible solution. A feasible solution is reached when all occurrence shifts are determined. Note that the nominal starting times (i.e. the starting times when no deviation occurs) can be determined by computing the longest path in the graph G where each arc (i, j) is valued by $p_i - \alpha H_{ij}$, and the adjustment is accomplished by shifting the starting of the following tasks by the value of the deviation. More details are provided in the next subsections.

3.1 Computation of an initial upper bound of the cycle time

In order to compute an initial upper bound, we design a heuristic that combines a greedy algorithm with a local search. The greedy algorithm assigns randomly a value to a given occurrence shift K_{ij} in the interval $[K_{ij}^-, 1-K_{ji}^-]$, and updates the bounds on the rest of the occurrences shifts such that the graph remains consistent. These two operations are repeated until all occurrence shifts are determined. Once all occurrence shifts are determined, a feasible schedule is obtained, consequently the associated optimal cycle time represents an upper bound of the global optimal cycle time. The local search algorithm consists in improving the cycle time by adjusting the values of the occurrence shifts that belong to the critical circuit. The idea behind these improvements is justified by the following proposition:

Proposition 1. *Let $(K_{ij})_{(i,j)\in\mathcal{D}}$ be a vector of feasible occurrence shifts and $\bar{\alpha}$ the associated cycle time given by the critical circuit c. Let $(u, v) \in \mathcal{D}$ be a disjunctive arc such that $(u, v) \in c$. If the following relation holds:*

$$\max_{l\in P^{uv}} \max_{p\in\mathcal{U}^\Gamma} \sum_{(i,j)\in l} p_i - \bar{\alpha} H_{ij} + p_v - \bar{\alpha}(K_{vu} - 1) \leq 0, \tag{14}$$

where P^{uv} is the set of paths from u to v, then the solution $(K'_{ij})_{(i,j)\in\mathcal{D}}$ where $K'_{uv} = K_{uv} + 1$ and $K'_{vu} = K_{vu} - 1$ has a cycle time less or equal to $\bar{\alpha}$.

Proof. Let $(K_{ij})_{(i,j)\in\mathcal{D}}$ be a vector of feasible occurrence shifts, $\bar{\alpha}$ the associated cycle time given by the critical circuit c and $(u, v) \in \mathcal{D}$ a disjunctive arc that belongs to c. Let us assume that relation (14) is verified. It is easily seen that putting $K'_{uv} = K_{uv} + 1$ makes the height of the circuit c increase by one and consequently makes decrease the value of its circuit ratio. In order to maintain the condition $K'_{uv} + K'_{vu} = 1$ verified, increasing the value of K_{uv} by one involve decreasing the value of K_{vu} by one. Now, it follows that decreasing the value of K_{vu} by one must ensure that the values of the circuits passing through the disjunctive arc (u, v) do not exceed $\bar{\alpha}$. This condition is verified, because by (14) we have:

$$\max_{l \in P^{uv}} \frac{\max\limits_{p \in \mathcal{U}^{\Gamma}} \sum_{(i,j) \in l} p_i + p_v}{\sum_{(i,j) \in l} H_{ij} + (K_{vu} - 1)} \leq \bar{\alpha}$$

In other words, the maximum circuit ratio passing by the disjunctive arc (j, i) has a value less or equal to $\bar{\alpha}$. Moreover, since the value of $\bar{\alpha}$ and the values of the processing times are positives, then $\sum_{(i,j) \in l} H_{ij} + (K_{vu} - 1) > 1$. This ensure that the associated graph to the robust CJSP is still consistent and the solution $(K^{'}_{ij})_{(i,j) \in \mathcal{D}}$ is feasible. $\square$

The pseudo-code of the proposed heuristic is given in Algorithm 1.

Algorithm 1 Initial upper bound computation

```
 1: Compute a lower bounds on the occurrences shifts $K_{ij}$;
 2: for all $(i, j) \in \mathcal{D}$ do
 3:     Update bounds on the occurrence shifts;
 4:     Affect randomly value to $K_{ij}$ on the interval $[K^{-}_{ij}, 1 - K^{-}_{ji}]$;
 5: end for
 6: Compute the associated cycle time $\bar{\alpha}$ and the critical circuit $c$.
 7: while $it < it_{max}$ do
 8:     Let $(u, v) \in \{(u, v) \in \mathcal{D}$ such that $(u, v) \in c\}$;
 9:     $l_{uv} \leftarrow \max_{l \in P^{uv}} \max_{p \in \mathcal{U}^{\Gamma}} \sum_{(i,j) \in l} p_i - \bar{\alpha} H_{ij}$;
10:     if $l_{uv} + p_v - \bar{\alpha}(K_{vu} - 1) \leq 0$ then
11:         $K_{uv} \leftarrow K_{uv} + 1$;
12:         $K_{vu} \leftarrow K_{vu} - 1$;
13:     end if
14:     Compute the associated cycle time $\bar{\alpha}$ and the critical circuit $c$;
15:     it$\leftarrow$ it+1;
16: end while
```

3.2 Lower bound

In the Branch-and-Bound algorithm, an initial lower bound is derived and compared to the incumbent. If the value of the initial lower bound and the value of the incumbent are equal, then an optimal solution is obtained and the Branch-and-Bound is stopped. It is easily seen that the problem where the disjunctive arcs are ignored is a relaxation of the initial problem. Consequently, the associated cycle time, α_{basic}, is a lower bound on the optimal cycle time. Furthermore, an other lower bound can be computed by reasoning on the machine charges. Let $M_{(i)} \in \mathcal{M}$ be a given machine and $S \subseteq \mathcal{T}$ the set of operations mapped on the machine $M_{(i)}$, then the optimal cycle time $\alpha_{opt} \geq \sum_{i \in S} p_i$, for each $p \in \mathcal{U}^{\Gamma}$.

Since this relation is verified for each machine, one can deduce the following lower bound:

$$\alpha_{machine} = \max_{m \in \mathcal{M}, p \in \mathcal{U}^{\Gamma}} \left\{ \sum_{i \in \mathcal{T} : M_{(i)} = m} p_i \right\}.$$

In the Branch-and-Bound procedure, we set the initial lower bound LB to the maximum value between $\alpha_{machine}$ and α_{basic}.

3.3 Node evaluation

In the Branch-and-Bound algorithm, we aim to find a feasible vector $(K_{ij})_{(i,j) \in \mathcal{D}}$ of occurrence shifts such that the value of the associated cycle time that ensure, for each $p \in \mathcal{U}^{\Gamma}$, the existence of schedule is minimum. In order to fathom nodes with partial solution in the search tree, it has to be evaluated by computing an associated lower bound. Let us consider a given node of the search tree defined by the subgraph $G_s = (\mathcal{T}, \mathcal{P} \cup \mathcal{D}_s)$, where $\mathcal{D}_s \subseteq \mathcal{D}$ is the set of fixed occurrence shifts. This subgraph represents a relaxation of the initial problem since only a subset a disjunctive arcs is considered. Consequently, the associate cycle time is a lower bound on the optimal cycle time.

3.4 Branching scheme and branching rule

To our knowledge, two branching schemes have been proposed for the cyclic job shop problem. In both of the branching schemes, the branching is performed on the unfixed occurrence shifts. The first one is introduced in [2]. Based on the interval of possibles values $[K^-_{ij}, 1 - K^-_{ji}]$ for the occurrence shift K_{ij} such that $(i, j) \in \mathcal{D}$, the author uses a dichotomic branching. In the first generated node, the interval of possible values of the occurrence shifts K_{ij} is restricted to $[K^-_{ij}, c_{ij}]$ and in the second one it is restricted to $[c_{ij} + 1, 1 - K^-_{ji}]$, where c_{ij} is the middle of the initial interval. The second branching scheme is introduced in [11]. The branching consists in selecting an unfixed disjunction and generate a child node for each possible value of the occurrence shift K_{ij} in the interval $[K^-_{ij}, 1 - K^-_{ji}]$. In each node, the algorithm assigns the corresponding possible value to the occurrence shift K_{ij}. In this paper, we follow the same branching scheme introduced in [11]. This branching scheme allows us to have, at each node, a subproblem which corresponds to a robust BCSP. Consequently, we can use the existing robust version of the Howard's algorithm to find the cycle time ensuring, for each $p \in \mathcal{U}^{\Gamma}$, the existence of a schedule. Different branching rules have been tested and numerical tests show that branching on occurrence shifts K_{ij} where $K^-_{ij} + K^-_{ji}$ is maximum yields best running times. This performance can be explained by the fact that this branching rule generates a small number of child nodes, which limits the size of the search tree.

4 Numerical experiments

We implemented the Branch-and-Bound algorithm in C++ and conducted the numerical experiments on an Intel Xeon E5-2695 processor running at 2.30GHz CPU. The time limit for each instance is set up to 900 seconds.

Since there are no existing benchmarks for the CJSP, even in its deterministic setting, we generate randomly 20 instances for each configuration as follows. We consider instances where the number of tasks n varies $\{10, 20, 30, 40, 50, 60, 80, 100\}$, the number of jobs j in $\{2, 3, 4, 5, 6, 10, 16\}$ and the number of machines m in $\{5, 6, 8, 10\}$. Each nominal duration $\bar{p}_i$ of task i is generated with uniform distribution in $[1, 10]$ and its deviation value $\hat{p}_i$ in $[0, 0.5\bar{p}_i]$.

Table 1 reports average solution times for the instances having from 10 to 40 tasks and with a budget of uncertainty varying from 0% to 100%. All these instances are solved before reaching the time limit. The average running times show that the Branch-and-Bound algorithm is not very sensitive to the variation of the budget of the uncertainty, but there is still a small difference. This can be explained by the number of the nodes explored in the Branch-and-Bound tree which can differ from an instance with a given value of Γ to another one. Table 1 also displays the percentage of deviation, for a given budget of uncertainty Γ, of the optimal cycle time α_{Γ} from the nominal optimal cycle time α_{nom}, where all tasks take their nominal values. This percentage of deviation is computed as $\text{Dev}_{\alpha} = \frac{\alpha_{\Gamma} - \alpha_{nom}}{\alpha_{nom}}$. The table shows that the percentage of deviations varies from 25.41% to 56.43%. In other words, these deviations represent the percentage of the nominal cycle time that has to be increased in order to protect a schedule against the uncertainty. We remark that the deviations stabilize when the budget of the uncertainty is greater than 20 or 30 percent. This situation occurs probably when the number of arcs of the circuit having the maximum number of arcs is less than Γ. In this case, increasing Γ does not influence the optimal cycle time. The second situation occurs when heights of other circuits than the actual critical circuit c have greater value than the height of c. In this case, increasing the budget of the uncertainty does not make the value of c lower than the others.

Table 2 shows the number of the instances that are solved before reaching the time limit. These results concern instances having from 50 to 100 tasks. The table shows that the Branch-and-Bound is not able to solve all these instances in less then 900 seconds. For example, among instances with 80 tasks, 16 jobs and 5 machines, only three instances have been solved.

5 Concluding remarks and perspectives

In this paper, we consider the cyclic job shop problem where the task durations are subject to uncertainty and belong to a polyhedral uncertainty set. We model the problem as two-stage robust optimization problem where the cycle time and the execution order of tasks mapped on the same machine have to be decided before knowing the uncertainty, and the starting times of tasks have to be determined after the uncertainty is revealed. We propose a Branch-and-Bound method that solves instances with at most 40 tasks but starts to have

# Tasks	# Jobs	# Machines	Γ(%)	Dev_α(%)	Time (s)
10	2	5	0	0	0.012
			10	25.41	0.0123
			20	41.89	0.0137
			30	48.95	0.0157
			40	51.67	0.0171
			50	53.18	0.0185
			70	53.49	0.0214
			90	53.49	0.0251
			100	53.49	0.0256
20	3	6	0	0	0.2980
			10	33.61	0.2136
			20	49.49	0.2432
			30	55.44	0.5685
			40	56.43	0.2994
			50	56.43	0.3258
			70	56.43	0.3783
			90	56.43	0.3996
			100	56.43	0.4695
30	5	8	0	0	22.6434
			10	38.25	12.0085
			20	49.03	15.4099
			30	50.91	66.2474
			40	50.91	16.3096
			50	50.91	17.1160
			70	50.91	13.8331
			90	50.91	15.7524
			100	50.91	15.8249
40	4	8	0	0	138.9174
			10	37.92	55.9220
			20	54.46	92.1455
			30	54.91	96.7587
			40	54.91	134.4442
			50	54.91	155.2327
			70	54.91	187.2088
			90	54.91	204.4813
			100	54.91	177.2455

Table 1. Average solution times in seconds for the Branch-and-Bound algorithm and percentage value of the deviation of the cycle time from the nominal cycle.

# tasks	# jobs	# machines	$\Gamma(\%)$								
			0	10	20	30	40	50	70	90	100
50	5	10	11	10	11	14	13	13	12	12	12
60	6	10	7	5	4	4	4	3	3	1	1
80	16	5	3	0	0	0	0	0	0	0	0
100	10	10	3	1	3	1	0	0	0	0	0

Table 2. Number of solved instances in less than 900 seconds among 20 instances.

difficulties with bigger instances. The next step is to investigate other techniques such as decomposition algorithms.

References

1. Roundy, R.: Cyclic schedules for job shops with identical jobs. Mathematics of operations research **17**(4) (1992) 842–865
2. Hanen, C.: Study of a np-hard cyclic scheduling problem: The recurrent job-shop. European journal of operational research **72**(1) (1994) 82–101
3. Brucker, P., Kampmeyer, T.: A general model for cyclic machine scheduling problems. Discrete Applied Mathematics **156**(13) (2008) 2561–2572
4. Che, A., Feng, J., Chen, H., Chu, C.: Robust optimization for the cyclic hoist scheduling problem. European Journal of Operational Research **240**(3) (2015) 627–636
5. Ben-Tal, A., Goryashko, A., Guslitzer, E., Nemirovski, A.: Adjustable robust solutions of uncertain linear programs. Mathematical Programming **99**(2) (2004) 351–376
6. Thiele, A., Terry, T., Epelman, M.: Robust linear optimization with recourse. Rapport technique (2009) 4–37
7. Zeng, B., Zhao, L.: Solving two-stage robust optimization problems using a column-and-constraint generation method. Operations Research Letters **41**(5) (2013) 457–461
8. Ayoub, J., Poss, M.: Decomposition for adjustable robust linear optimization subject to uncertainty polytope. Computational Management Science **13**(2) (2016) 219–239
9. Gondran, M., Minoux, M., Vajda, S.: Graphs and Algorithms. John Wiley & Sons, Inc., New York, NY, USA (1984)
10. Dasdan, A.: Experimental analysis of the fastest optimum cycle ratio and mean algorithms. ACM Transactions on Design Automation of Electronic Systems (TODAES) **9**(4) (2004) 385–418
11. Fink, M., Rahhou, T.B., Houssin, L.: A new procedure for the cyclic job shop problem. IFAC Proceedings Volumes **45**(6) (2012) 69–74
12. Bertsimas, D., Sim, M.: The price of robustness. Operations research **52**(1) (2004) 35–53
13. Hamaz, I., Houssin, L., Cafieri, S.: Robust Basic Cyclic Scheduling Problem. Technical report (2017)