Preprints of the 13th IFAC Symposium on Information Control Problems in Manufacturing, Moscow, Russia, June 3 - 5, 2009

Partial job order for solving the two-machine flow-shop minimum-length problem with uncertain processing times

Natalia M. Matsveichuk * **Yuri N. Sotskov** ** **Frank Werner** ***

* *United Institute of Informatics Problems, Minsk, 220012 Belarus (Tel: 375-17-2842125; e-mail: leshchenko@newman.bas-net.by).*
** *United Institute of Informatics Problems, Minsk, 220012 Belarus (Tel: 375-17-2842120; e-mail: sotskov@newman.bas-net.by).*
*** *Fakultät für Mathematik, Otto-von-Guericke-Universität, Magdeburg, 39106 Germany (Tel: +49 391 6712025; frank.werner@mathematik.uni-magdeburg.de)*

Abstract: The flow-shop minimum-length scheduling problem with n jobs processed on two machines is addressed where processing times are uncertain (only lower and upper bounds for the random processing time are given, while the probability distribution between these bounds is unknown). For such a problem, there often does not exist a dominant schedule that remains optimal for all possible realizations of the job processing times, and so we look for a minimal dominant set of schedules, which may be represented by a partial job order. We investigate properties of this partial job order and show how to construct this order in polynomial time. The approach based on a set of dominant schedules allows us to find special cases of the problem when it is possible to find an optimal schedule in spite of the uncertainty of the numerical data.

Keywords: Scheduling; Flow-shop; Makespan; Uncertainty; Domination

1. INTRODUCTION

In the scheduling literature, the processing times are mainly assumed to be either *deterministic* values (see part 1 of the book Pinedo [1995]) or *random* variables with a given probability distribution (see part 2 of Pinedo [1995]). Unfortunately, in many real-life scheduling situations, one may have no sufficient information to characterize the probability distribution for each processing time. In this paper, we consider the two-machine minimum-length flow-shop scheduling problem with uncertain processing times.

Let two machines $\mathcal{M} = \{M_1, M_2\}$ be given to process $n \geq 2$ jobs $\mathcal{J} = \{J_1, J_2, ..., J_n\}$ that have to follow the same machine route: Each job $J_i \in \mathcal{J}$ has to be processed by machine M_1 and then by machine M_2 without preemption on each machine. All the n jobs are available to be processed from time $t_0 = 0$. Let $C_i(\pi)$ denote the completion time of job $J_i \in \mathcal{J}$ in schedule π, and the criterion C_{max} denotes the minimization of the schedule length $C_{max}(\pi)$:

$$C_{max} = \min_{\pi \in \Omega} C_{max}(\pi) = \min_{\pi \in \Omega} \{\max\{C_i(\pi) \mid J_i \in \mathcal{J}\}\},$$

where Ω is a set of semi-active schedules with cardinality $|\Omega| = (n!)^2$ (Tanaev et al. [1994]). In a semi-active schedule, the processing of each job $J_i \in \mathcal{J}$ starts as early as possible (provided that the order of the jobs $\mathcal{J}$ for processing is fixed). In contrast to the conventional two-machine minimum-length flow-shop problem $F2||C_{max}$ with fixed processing times (see Pinedo [1995], Tanaev et al. [1994]), it is assumed that the processing time p_{ij} of job $J_i \in \mathcal{J}$ on machine $M_j \in \mathcal{M}$ is unknown before scheduling. In the realization of the process, p_{ij} may take any real value in the segment $[p_{ij}^L,\ p_{ij}^U]$, where the lower bound p_{ij}^L and the upper bound p_{ij}^U are fixed, but the probability distributions of the processing times between these bounds are unknown. Such a two-machine minimum-length flow-shop scheduling problem with uncertain processing times is denoted by $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$.

2. PRELIMINARIES

Let $S = \{\pi_1, \pi_2, \ldots, \pi_{n!}\}$ be the set of all permutations of n jobs from set $\mathcal{J}$:

$$\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}),\ \{k_1, k_2, \ldots, k_n\} = \{1, 2, \ldots, n\}.$$

The set S defines all *permutation schedules* that *dominate* the set Ω of semi-active schedules for problem $F2||C_{max}$: There exists at least one optimal semi-active schedule with the same sequence of jobs on both machines M_1 and M_2 (see Johnson [1954], Tanaev et al. [1994]). Since each permutation $\pi_k \in S$ uniquely defines the set of *earliest* completion times $C_i(\pi_k)$ of the jobs $J_i \in \mathcal{J}$ for problem $F2||C_{max}$, we identify a *permutation* $\pi_k \in S$ with a *permutation schedule* defined by π_k. Let $T = \{p \mid p_{ij}^L \leq p_{ij} \leq p_{ij}^U,\ J_i \in \mathcal{J},\ M_j \in \mathcal{M}\}$ be the set of possible vectors $p = (p_{1,1}, p_{1,2}, \ldots, p_{n1}, p_{n2})$ of the job processing times.

For a fixed vector $p \in T$, problem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$ turns into the problem $F2||C_{max}$ associated with the vector p of job processing times, and so for each vector $p \in T$, it is sufficient to look for an optimal schedule among the set S of permutation schedules. Thus, the set S of permutation schedules is *dominant* for problem $F2|p_{ij}^L \leq$

$p_{ij} \le p_{ij}^U | C_{max}$ as well. The set of permutation schedules has the cardinality $|S| = n!$. Next, we restrict further the set of permutations that are sufficient to examine while solving problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$.

Johnson [1954] proved that permutation $\pi_i = (J_{i_1}, J_{i_2}, \ldots, J_{i_n}) \in S$ (called a Johnson permutation) satisfying the condition

$$\min\{p_{i_k 1}, p_{i_m 2}\} \le \min\{p_{i_m 1}, p_{i_k 2}\}, \quad (1)$$

$1 \le k < m \le n$, is optimal for problem $F2||C_{max}$. In contrast to problem $F2||C_{max}$ with fixed processing times, for problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ with uncertain processing times there often does not exist a single permutation of the n jobs from $\mathcal{J}$ that remains optimal for all possible realizations of the job processing times. So a *minimal dominant set* of permutations has to be treated as a solution to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$.

Definition 1. The set of permutations $S(T) \subseteq S$ is called a J-solution to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$, if for each vector $p \in T$, the set $S(T)$ contains at least one permutation that is a Johnson one for problem $F2||C_{max}$ associated with the vector p of job processing times. Any proper subset of set $S(T)$ is not a J-solution to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$.

From Definition 1, it follows that set $S(T)$ contains at least one optimal schedule $\pi_k \in S(T) \subseteq S$ for each vector $p \in T$ of the job processing times and that set $S(T)$ is a minimal set (with respect to inclusion) which possesses such a property. Thus, to solve problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$, one can restrict the search within the set $S(T)$ which often has an essentially smaller cardinality than set S.

Braun et al. [2002] (Braun et al. [2006]) used the stability of a Johnson (Jackson) permutation for solving a two-machine minimum-length flow-shop (job-shop) scheduling problem with limited machine availability. A minimal dominant set of schedules was investigated by Allahverdi and Sotskov [2003], Lai and Sotskov [1999], Lai et al. [1997] for the C_{max} criterion, and by Allahverdi et al. [2003], Lai et al. [2004], Sotskov et al. [2004] for the total flow time criterion. In particular, Allahverdi and Sotskov [2003] identified the following sufficient conditions (2) and (3) for fixing the order $J_v \to J_w$ of the jobs $J_v \in \mathcal{J}$ and $J_w \in \mathcal{J}$ while solving problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$.

If

$$p_{v1}^U \le p_{v2}^L \text{ and } p_{v1}^U \le p_{w1}^L, \quad (2)$$

then for each vector $p \in T$, there exists a permutation $\pi_k = (s_1, J_v, s_2, J_w, s_3) \in S$ that is a Johnson one for problem $F2||C_{max}$ associated with the vector p of job processing times.

If

$$p_{w2}^U \le p_{w1}^L \text{ and } p_{w2}^U \le p_{v2}^L, \quad (3)$$

then for each vector $p \in T$, there exists a permutation $\pi_k = (s_1, J_v, s_2, J_w, s_3) \in S$ that is a Johnson one for problem $F2||C_{max}$ associated with the vector p of job processing times.

Leshchenko and Sotskov [2005] proved that, if both conditions (2) and (3) do not hold, then there is no J-solution $S(T)$ with the same order $J_v \to J_w$ in all permutations $\pi_i \in S(T)$. Summarizing, the following claim has been proven.

Table 1. Lower and upper bounds for the possible processing times for Example 1

J_i	J_1	J_2	J_3	J_4	J_5	J_6	J_7
p_{i1}^L	2	1	4	6	9	7	4
p_{i1}^U	5	3	8	8	10	9	5
p_{i2}^L	8	10	6	7	9	6	2
p_{i2}^U	9	12	7	7	10	8	3

Theorem 1. There exists a J-solution $S(T)$ to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ with the fixed order $J_v \to J_w$ of jobs $J_v \in \mathcal{J}$ and $J_w \in \mathcal{J}$ in all permutations $\pi_k \in S(T)$ if and only if at least one condition (2) or (3) holds.

3. A BINARY RELATION DEFINED BY SET $S(T)$

Let $\mathcal{J} \times \mathcal{J}$ denote the Cartesian product of set $\mathcal{J}$. Due to Theorem 1, by testing inequalities (2) and (3) for each pair of jobs $J_v \in \mathcal{J}$ and $J_w \in \mathcal{J}$, we can construct the following binary relation $\mathcal{A}_{\preceq} \subseteq \mathcal{J} \times \mathcal{J}$ on set $\mathcal{J}$: Inclusion $(J_v, J_w) \in \mathcal{A}_{\preceq}$ holds, $v \ne w$, if and only if there exists a J-solution $S(T)$ to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ such that job $J_v \in \mathcal{J}$ is located before job $J_w \in \mathcal{J}$ (i.e., $J_v \to J_w$) in all permutations $\pi_k \in S(T)$.

The binary relation $\mathcal{A}_{\preceq}$ defines a digraph $(\mathcal{J}, \mathcal{A}_{\preceq})$ with vertex set $\mathcal{J}$ and arc set $\mathcal{A}_{\preceq}$. Relation $(J_v, J_w) \in \mathcal{A}_{\preceq}$ will also be represented as follows: $J_v \preceq J_w$. It is clear that it takes $O(n^2)$ time to construct the digraph $(\mathcal{J}, \mathcal{A}_{\preceq})$ by testing inequalities (2) and (3) for each pair of jobs from set $\mathcal{J}$. In the following, we consider the partition $\mathcal{J} = \mathcal{J}_0 \cup \mathcal{J}_1 \cup \mathcal{J}_2 \cup \mathcal{J}^*$, where $\mathcal{J}_0 = \{J_i \in \mathcal{J} \mid p_{i1}^U \le p_{i2}^L, p_{i2}^U \le p_{i1}^L\}$, $\mathcal{J}_1 = \{J_i \in \mathcal{J} \mid p_{i1}^U \le p_{i2}^L, p_{i2}^U > p_{i1}^L\} = \{J_i \in \mathcal{J} \setminus \mathcal{J}_0 \mid p_{i1}^U \le p_{i2}^L\}$, $\mathcal{J}_2 = \{J_i \in \mathcal{J} \mid p_{i1}^U > p_{i2}^L, p_{i2}^U \le p_{i1}^L\} = \{J_i \in \mathcal{J} \setminus \mathcal{J}_0 \mid p_{i2}^U \le p_{i1}^L\}$, $\mathcal{J}^* = \{J_i \in \mathcal{J} \mid p_{i1}^U > p_{i2}^L, p_{i2}^U > p_{i1}^L\}$. For each job $J_k \in \mathcal{J}_0$, from the inequalities $p_{k1}^U \le p_{k2}^L$ and $p_{k2}^U \le p_{k1}^L$, we obtain the equalities $p_{k1}^L = p_{k1}^U = p_{k2}^L = p_{k2}^U$. Thus, the processing times p_{k1} and p_{k2} are fixed and equal for both machines: $p_{k1} = p_{k2} = p_k$.

For illustration, we consider Example 1 of problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ with the job set $\mathcal{J} = \{J_1, J_2, J_3, J_4, J_5, J_6, J_7\}$, where $\mathcal{J}_1 = \{J_1, J_2\}$, $\mathcal{J}^* = \{J_3, J_4, J_5, J_6\}$ and $\mathcal{J}_2 = \{J_7\}$. The segments of the possible processing times are given in Table 1. Due to Theorem 1, by testing inequalities (2) and (3) for each pair of jobs from set $\mathcal{J} = \{J_1, J_2, J_3, J_4, J_5, J_6, J_7\}$, we can construct the binary relation $\mathcal{A}_{\preceq}$ on set $\mathcal{J}$, represented in Fig. 1.

In the general case, the binary relation $\mathcal{A}_{\preceq}$ may be not transitive. However, we have proved the following claim.

Theorem 2. If $\mathcal{J}_0 = \emptyset$, then the binary relation $\mathcal{A}_{\preceq}$ is transitive.

To illustrate the non-transitivity of a binary relation $\mathcal{A}_{\preceq}$ in the case $\mathcal{J}_0 \ne \emptyset$, we consider Example 2 of problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ with the job set $\mathcal{J} = \{J_1, J_2, J_3\}$, where $J_1 \in \mathcal{J}^*$, $J_2 \in \mathcal{J}^*$ and $J_3 \in \mathcal{J}_0$. Let the following conditions hold:

$$p_{1,1}^L < p_3 < p_{1,2}^L, \quad (4)$$

$$p_{2,2}^L < p_3 < p_{2,1}^L. \quad (5)$$

Due to Theorem 1, by testing inequalities (2) and (3) for each pair of jobs, one can construct the following binary relation $\mathcal{A}_{\preceq}$ on the set $\mathcal{J} = \{J_1, J_2, J_3\}$, where $J_1 \preceq J_3$ and $J_3 \preceq J_2$. Due to inequalities (4) and (5), neither inequality (2) nor inequality (3) holds for $v = 1$ and $w = 2$ and for $v = 2$ and $w = 1$. Thus, we obtain $J_1 \not\preceq J_2$ and $J_2 \not\preceq J_1$, and the binary relation $\mathcal{A}_{\preceq}$ is not transitive (see Fig. 2).

In what follows, only the case $\mathcal{J}_0 = \emptyset$ will be considered.

Let $\mathcal{J}_0 = \emptyset$, i.e., $\mathcal{J} = \mathcal{J}^* \cup \mathcal{J}_1 \cup \mathcal{J}_2$. For the jobs $J_v \in \mathcal{J}_1$ and $J_w \in \mathcal{J}_1$, it may happen that there exist both a J-solution $S(T) \subset S$ with job J_v located before job J_w (i.e., $J_v \to J_w$) in all permutations $\pi_k \in S(T)$ and a J-solution $S'(T) \subset S$ with job J_w located before job J_v (i.e., $J_w \to J_v$) in all permutations $\pi_l \in S'(T)$. In such a case, digraph $(\mathcal{J}, \mathcal{A}_{\preceq})$ has a circuit (J_v, J_w, J_v).

We consider the following Example 3 of problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ which shows the occurrence of a circuit in the digraph $(\mathcal{J}, \mathcal{A}_{\preceq})$ with the job set $\mathcal{J} = \{J_1, J_2\}$.

Let $J_1 \in \mathcal{J}_2$, $J_2 \in \mathcal{J}_2$ and condition

$$p_{1,2}^L = p_{1,2}^U = p_{2,2}^L = p_{2,2}^U \tag{6}$$

hold. One of the possible realizations of equalities (6) is shown in Fig. 3.

Due to Theorem 1, by testing inequalities (2) and (3) for $v = 1$ and $w = 2$ and for $v = 2$ and $w = 1$, one can construct the following binary relation $\mathcal{A}_{\preceq}$ on the set $\mathcal{J}$: $J_1 \preceq J_2$ and $J_2 \preceq J_1$. Therefore, the binary relation $\mathcal{A}_{\preceq}$ contains a circuit (J_1, J_2, J_1) (see Fig. 4).

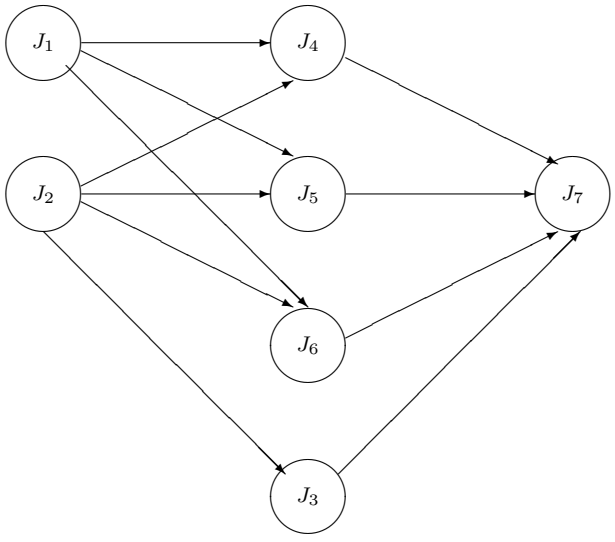

Fig. 1. Digraph $(\mathcal{J}, \mathcal{A}_{\preceq})$ (transitive arcs are omitted) representing the binary relation $\mathcal{A}_{\preceq}$ on the set $\mathcal{J}$ for Example 1.

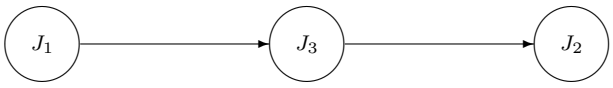

Fig. 2. Non-transitivity of the binary relation $\mathcal{A}_{\preceq}$ on the set $\mathcal{J} = \{J_1, J_2, J_3\}$ for Example 2 with $\mathcal{J}_0 \neq \emptyset$.

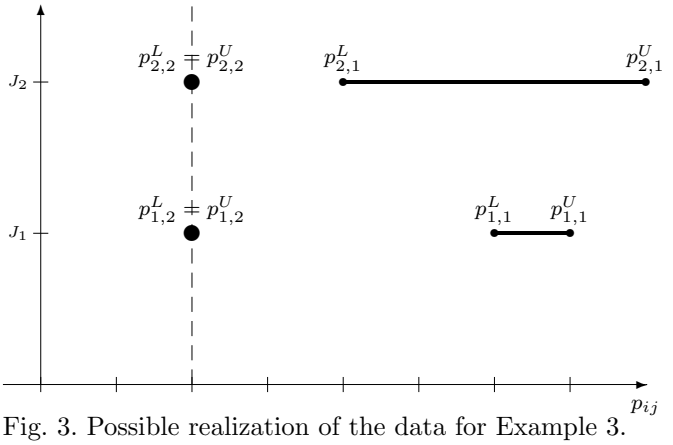

Fig. 3. Possible realization of the data for Example 3.

Theorem 3. Digraph $(\mathcal{J}, \mathcal{A}_{\preceq})$ has no circuits if and only if set $\mathcal{J} = \mathcal{J}^* \cup \mathcal{J}_1 \cup \mathcal{J}_2$ includes no pair of jobs $J_i \in \mathcal{J}_k$ and $J_j \in \mathcal{J}_k$ with $k \in \{1, 2\}$ such that the following equalities hold:

$$p_{ik}^L = p_{ik}^U = p_{jk}^L = p_{jk}^U. \tag{7}$$

In the case $\mathcal{J}_0 = \emptyset$ under consideration, we define a binary relation $\mathcal{A}_{\prec} \subseteq \mathcal{A}_{\preceq} \subseteq \mathcal{J} \times \mathcal{J}$ using the following agreement: If $J_v \preceq J_w$ and $J_w \not\preceq J_v$, then $J_v \prec J_w$. If $J_v \preceq J_w$ and $J_w \preceq J_v$ with $v < w$, then $J_v \prec J_w$ and $J_w \not\prec J_v$.

Since set $\mathcal{J}_0$ is empty, we obtain an antireflective, antisymmetric (due to Theorem 3), and transitive (due to Theorem 2) binary relation $\mathcal{A}_{\prec}$ on the set $\mathcal{J}$, i.e., we obtain a strict order. The strict order $\mathcal{A}_{\prec}$ defines a digraph $\mathcal{G} = (\mathcal{J}, \mathcal{A}_{\prec})$ with the vertex set $\mathcal{J}$ and the arc set $\mathcal{A}_{\prec}$. We have proved that all components of digraph $\mathcal{G}$ (except at most one) are isolated vertices.

Theorem 4. Let $\mathcal{J} = \mathcal{J}^* \cup \mathcal{J}_1 \cup \mathcal{J}_2$. Then there exists at most one component with a cardinality greater than one in the digraph $\mathcal{G}$.

A permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$ may be considered as a total order of the jobs $\mathcal{J}$. A total order defined by permutation π_k is called a *linear extension* of a partial order $\mathcal{A}_{\prec}$, if each inclusion $(J_{k_u}, J_{k_v}) \in \mathcal{A}_{\prec}$ implies inequality $u < v$. Let $\Pi(\mathcal{G})$ denote the set of permutations $\pi_k \in S$ defining all linear extensions of the partial order $\mathcal{A}_{\prec}$. In particular, if $\mathcal{G} = (\mathcal{J}, \emptyset)$, then $\Pi(\mathcal{G}) = S$. On the other hand, if $|\mathcal{A}_{\prec}| = \frac{n(n-1)}{2}$, then $\Pi(\mathcal{G}) = \{\pi_k\}$. Necessary and sufficient conditions for the equality $\Pi(\mathcal{G}) = \{\pi_k\}$ have been proved by Leshchenko and Sotskov [2005]. We have proved the following claim.

Theorem 5. Let $\mathcal{J} = \mathcal{J}^* \cup \mathcal{J}_1 \cup \mathcal{J}_2$. For any vector $p \in T$, set $\Pi(\mathcal{G})$ contains a Johnson permutation for problem $F2||C_{max}$ associated with the vector p of job processing times.

Due to Theorem 5, set $\Pi(\mathcal{G})$ satisfies the first part of Definition 1. However, this set may violate the second part of Definition 1 since it may include a *redundant*

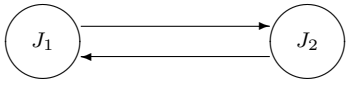

Fig. 4. Digraph $(\mathcal{J}, \mathcal{A}_{\preceq})$ constructed for Example 3.

permutation π_k, i.e., set $\Pi(\mathcal{G})\backslash\{\pi_k\}$ may also include at least one Johnson permutation for any vector $p \in T$ of the job processing times. Obviously, after deleting from set $\Pi(\mathcal{G})$ all such redundant permutations, we obtain a set $\Pi^*(\mathcal{G})$ which coincides with a minimal dominant set $S(T)$. Thus, the following claim can be proven.

Corollary 1. If $\mathcal{J} = \mathcal{J}^* \cup \mathcal{J}_1 \cup \mathcal{J}_2$, then there exists a J-solution $S(T)$ to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ such that $S(T) \subseteq \Pi(\mathcal{G})$.

In the next section, we show how to construct the set $\Pi^*(\mathcal{G}) = S(T)$.

4. HOW TO CONSTRUCT SET $S(T) \subseteq \Pi(\mathcal{G})$?

Let the pair of jobs $J_i \in \mathcal{J}$ and $J_j \in \mathcal{J}$ be called a *conflict pair* of jobs, if neither relation $J_i \preceq J_j$ nor relation $J_j \preceq J_i$ holds. We have proved the following three lemmas.

Lemma 1. Let $\mathcal{J} = \mathcal{J}^* \cup \mathcal{J}_1 \cup \mathcal{J}_2$. A permutation of the form $\pi_g = (\ldots, J_i, \ldots, J_r, \ldots, J_j, \ldots) \in \Pi(\mathcal{G})$ is redundant, if for a pair of jobs $J_i \in \mathcal{J}_k$ and $J_j \in \mathcal{J}_k$ with $k \in \{1, 2\}$ equalities (7) hold and job J_r creates a conflict pair both with job J_i and with job J_j.

A redundant permutation defined in Lemma 1 will be called a *redundant permutation of type 1.* Due to Lemma 1, testing whether set $\Pi(\mathcal{G})$ contains a redundant permutation of type 1 takes $O(n^2)$ time. In order to delete all redundant permutations of type 1 from set $\Pi(\mathcal{G})$, it is sufficient to identify each pair of jobs J_i and J_j in digraph $\mathcal{G}$ for which the condition of Lemma 1 holds. If $\mathcal{G}^I$ is the digraph obtained from digraph $\mathcal{G} = (\mathcal{J}, \mathcal{A}_{\prec})$ after such an identification of vertices in the set $\mathcal{J}$, then set $\Pi(\mathcal{G}^I)$ has no redundant permutations of type 1. Let inclusion $J_j \in \mathcal{J}^*$ hold. We define two sets of jobs as follows:

$$\mathcal{J}_j' = \{J_q \in \mathcal{J}_2 \mid \min\{p_{j1}^U, p_{j2}^U\} < p_{q2}^U\} \bigcup \{J_r \in \mathcal{J}_1 \cup \mathcal{J}^* \mid \min\{p_{j1}^U, p_{j2}^U\} \le p_{r1}^L\};$$

$$\mathcal{J}_j'' = \{J_w \in \mathcal{J}_1 \mid \min\{p_{j1}^U, p_{j2}^U\} < p_{w1}^U\} \bigcup \{J_u \in \mathcal{J}_2 \cup \mathcal{J}^* \mid \min\{p_{j1}^U, p_{j2}^U\} \le p_{u2}^L\}.$$

Lemma 2. Let $\mathcal{J} = \mathcal{J}^* \cup \mathcal{J}_1 \cup \mathcal{J}_2$. If $J_j \in \mathcal{J}^*$, $J_q \in \mathcal{J}_j'$, $J_w \in \mathcal{J}_j''$, then each permutation of the form $\pi_g = (\ldots, J_q, \ldots, J_j, \ldots, J_w, \ldots) \in \Pi(\mathcal{G})$ is redundant.

A redundant permutation defined by the condition of Lemma 2 will be called a *redundant permutation of type 2.* Due to Lemma 2, testing whether permutation $\pi_g \in \Pi(\mathcal{G})$ is a redundant permutation of type 2 takes $O(n)$ time.

Let $\Pi^*(\mathcal{G})$ denote the set of permutations left in the set $\Pi(\mathcal{G})$ after deleting all redundant permutations of type 1 and type 2.

Lemma 3. Let $\mathcal{J} = \mathcal{J}^* \cup \mathcal{J}_1 \cup \mathcal{J}_2$. If permutation $\pi_t \in \Pi(\mathcal{G})$ is redundant in set $\Pi(\mathcal{G})$, then π_t is a redundant permutation either of type 1 or type 2.

Using Lemmas 1 - 3 and Theorem 5, the following claim may be proven.

Theorem 6. If set $\mathcal{J} = \mathcal{J}^* \cup \mathcal{J}_1 \cup \mathcal{J}_2$ does not contain a pair of jobs $J_i \in \mathcal{J}_k$ and $J_j \in \mathcal{J}_k$, $k \in \{1, 2\}$, such that condition

$$\max\{p_{i,3-k}^L, p_{j,3-k}^L\} < p_{ik}^L = p_{ik}^U = p_{jk}^L = p_{jk}^U < \min\{p_{i,3-k}^U, p_{j,3-k}^U\} \quad (8)$$

holds, then $\Pi^*(\mathcal{G}) = S(T)$.

It is clear that testing the condition of Theorem 6 takes $O(n)$ time. Due to Theorem 6 and Lemma 3, if set $\mathcal{J} = \mathcal{J}^* \cup \mathcal{J}_1 \cup \mathcal{J}_2$ does not contain a pair of jobs $J_i \in \mathcal{J}_k$ and $J_j \in \mathcal{J}_k$, $k \in \{1, 2\}$, such that condition (8) holds, then a J-solution may be constructed by deleting all redundant permutations of types 1 and 2 from set $\Pi(\mathcal{G})$. Since the obtained set $\Pi^*(\mathcal{G})$ is uniquely defined, the following claim is correct.

Corollary 2. If set $\mathcal{J} = \mathcal{J}^* \cup \mathcal{J}_1 \cup \mathcal{J}_2$ does not contain a pair of jobs J_i and J_j such that condition (8) holds, then relation $\mathcal{A}_{\prec}$ defines a unique J-solution $\Pi^*(\mathcal{G}) = S(T)$ to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$.

The condition of Theorem 6 is sufficient for the uniqueness of a J-solution $\Pi^*(\mathcal{G}) = S(T)$ to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$. However, this condition is not necessary for the uniqueness of a J-solution defined by relation $\mathcal{A}_{\prec}$ as demonstrated by the following Example 4.

Let us consider Example 4 of problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ with the job set $\mathcal{J} = \{J_1, J_2, J_3\}$, where $J_1 \in \mathcal{J}^*$, $J_2 \in \mathcal{J}^*$ and $J_3 \in \mathcal{J}_1$. Let the following conditions hold: $p_{1,2}^L < p_{1,1}^L = p_{1,1}^U < p_{1,2}^U$, $p_{2,2}^L < p_{2,1}^L = p_{2,1}^U < p_{2,2}^U$, $p_{1,1}^L = p_{2,1}^L$, $p_{1,1}^L \le p_{3,1}^L$, (see Fig. 5 for a possible realization of such input data).

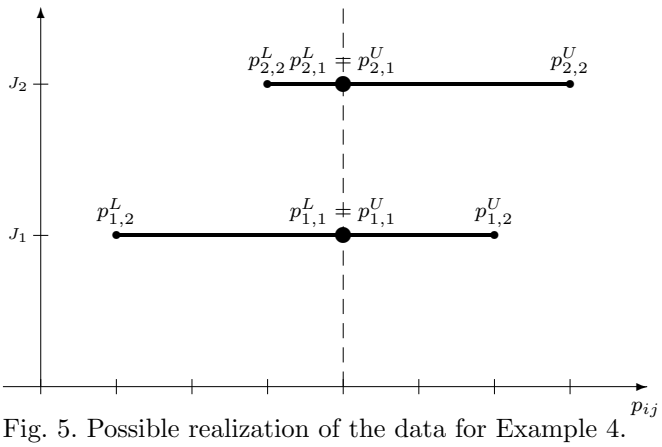

Fig. 5. Possible realization of the data for Example 4.

Due to Theorem 1, by testing inequalities (2) and (3) for each pair of jobs, one can construct a binary relation $\mathcal{A}_{\preceq}$ on the set $\mathcal{J}$ and the corresponding digraph $\mathcal{G}$. It is easy to see that for Example 4, digraph $\mathcal{G}$ has an empty set of arcs: $\mathcal{G} = (\mathcal{J}, \emptyset)$. Therefore, there are six permutations in the set $\Pi(\mathcal{G})$: $\pi_1 = (J_1, J_2, J_3)$, $\pi_2 = (J_1, J_3, J_2)$, $\pi_3 = (J_2, J_1, J_3)$, $\pi_4 = (J_2, J_3, J_1)$, $\pi_5 = (J_3, J_1, J_2)$, $\pi_6 = (J_3, J_2, J_1)$. The four permutations π_2, π_4, π_5 and π_6 are not redundant. Permutations π_1 and π_3 are redundant, however, every J-solution $S(T)$ has to contain one of these two permutations. Thus, there are two different J-solutions $S_1(T)$ and $S_2(T)$ for Example 4 of problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$: $S_1(T) = \{\pi_1, \pi_2, \pi_4, \pi_5, \pi_6\}$, $S_2(T) = \{\pi_2, \pi_3, \pi_4, \pi_5, \pi_6\}$. Note that after deleting job J_3, one can obtain Example 5 of problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ with two jobs, $\mathcal{J} = \{J_1, J_2\}$, and a unique J-solution $S(T) = \{\pi_1, \pi_2\}$, where $\pi_1 = (J_1, J_2)$, $\pi_2 = (J_2, J_1)$.

5. HOW TO USE THE ABOVE RESULTS?

The practical significance of the obtained results will be clear from the new approach for solving problem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$ which allows us to execute a schedule that becomes optimal for the actual processing times. In this section, we outline the method based on Theorems 1, 2, 3, 4, 5, 6 and Corollaries 1, 2 for dealing with uncertain numerical input data. A formal description of the corresponding algorithms along with computational results is given by Matsveichuk et al. [2009]. The approach based on constructing a minimal set of dominant schedules $S(T)$ allows us to find special cases of a problem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$ when it is possible to find an actual optimal schedule in spite of the uncertain numerical data. If $|S(T)| \geq 2$, then the solution process can be seen as consisting of a *static* and a *dynamic phase*. In the *static phase*, a scheduler can use Theorems 1, 2, 3, 4, 5, 6 and Corollaries 1, 2 to construct a J-solution $S(T)$ to problem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$ if $\mathcal{J}_0 = \emptyset$ (the case $\mathcal{J}_0 \neq \emptyset$ is considered by Matsveichuk et al. [2009]). In the *dynamic phase* of the decision-making process, a scheduler has to select an appropriate schedule from the set $S(T)$ to react in real-time to the actual processing times of the already completed jobs. A decision point t_i, $t_i > t_0 = 0$, may be equal to the actual completion time of job $J_i \in \mathcal{J}$ on machine M_1 if there exists a conflict pair of jobs at time t_i. At time $t_i \geq 0$, a scheduler can make the right decision to choose one of the conflict jobs to be processed next on machine M_1, if after this decision, the obtained subset of set $S(T)$ remains a J-solution to the remaining subproblem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$ with the corresponding set $T_i \subset T$ of the possible vectors of job processing times. Otherwise, a scheduler may use one of the solution policies that does not guarantee to find an optimal schedule for any realization of the remaining job processing times. The solution policy may be optimistic or pessimistic, or a scheduler can minimize the objective function C_{max} on average (see Aytug et al. [2005]), or a scheduler can choose arbitrarily one of the conflict jobs to be processed next.

In Sections 2, 3 and 4, we have developed the mathematical background for the *static phase* of scheduling (Theorems 1, 2, 3, 4, 5, 6 and Corollaries 1, 2). The *static phase* may be considered as predictive scheduling and the *dynamic phase* as reactive scheduling. After the *static phase* of scheduling, a predictive set of schedules $S(T)$ is constructed in the form of a digraph $(\mathcal{J}, A_{\prec})$. The digraph $(\mathcal{J}, A_{\preceq})$ constructed in the *static phase* characterizes a minimal dominant set $S(T)$ of schedules without enumerating them explicitly. In general, any schedule from the set $S(T)$ can be implemented as an optimal one during the *dynamic phase* for some suitable realizations of the job processing times. Matsveichuk et al. [2009] gave a detailed investigation of the *dynamic phase* of scheduling. Namely, a lot of sufficient conditions for the existence of a dominant permutation $\pi_u \in S(T)$ are proven provided that the initial part of the schedule is already executed and the processing times of the completed jobs become known exactly. Using these sufficient conditions (along with Theorems 1, 2, 3, 4, 5, 6 and Corollaries 1, 2), two-phase scheduling algorithms are developed and coded in C++. These algorithms fall into the category of *predictive-reactive* scheduling (Aytug et al. [2005]). In the *static phase* of scheduling, digraph $(\mathcal{J}, A_{\prec})$ is considered as a form of a schedule for problem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$. In the *dynamic phase*, the partial orders of the jobs defined by digraph $(\mathcal{J}, A_{\prec})$ have to be progressively extended using additional information about the processing times of the jobs being completed.

The page limit of a paper does not allow us to present here the details of these algorithms. It should be only noted that the sufficient conditions for a domination of a permutation proven for the *dynamic phase* do not exploit Johnson's condition (1) in contrast to Definition 1, Theorems 1, 2, 3, 4, 5, 6 and Corollaries 1, 2. Moreover, if no sufficient condition holds for a permutation of conflict jobs at a decision point t_i, then a job to be processed next on machine M_1 was selected arbitrarily from the set of all conflict jobs.

The preliminary computational results have shown the efficiency (in computational time) of these two-phase scheduling algorithms and their effectiveness (in the number of randomly generated instances solved exactly). In particular, if the relative error of the input data was not large (less than 10%), then most randomly generated problems $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$ with $50 \leq n \leq 5000$ have been solved exactly within 0.05 seconds per instance in spite of uncertain processing times. Moreover, the optimality of the actual schedule was often proven before its completion, and the average error of the objective function C_{max} was no more than 3%. For most randomly generated instances solved exactly in our experiments, the cardinality of set $S(T)$ was rather large. Nevertheless, the average CPU-time of a Celeron 1200 MHz processor for both the *static phase* and the *dynamic phase* of the decision-making process was less than 1.5 seconds even for instances with 10000 jobs.

If set $\mathcal{J}_0$ is empty, then digraph $G = (\mathcal{J}, A_{\prec})$ represents a unique solution $S(T)$ to problem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$ and may be considered as the union of optimal schedules constructed for the set of problems $F2||C_{max}$, which may be generated from the original problem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$ for different realizations of the job processing times. In the general case (if set $\mathcal{J}_0$ may be not empty), using solution $S(T)$ a scheduler has to look for an optimal schedule when some jobs from set $\mathcal{J}$ will be in process. In other words, it may be possible to arrange the jobs that cause a conflict in real-time when additional information about the job processing (completion) times becomes available for a scheduler. It is clear that, the larger the uncertainty of the input data is, the less possibility for the right decision a scheduler will have. The cardinality of set $S(T)$ may be considered as a *measure* of uncertainty of problem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$. With respect to Johnson's algorithm, jobs from the sets $\mathcal{J}_1$ and $\mathcal{J}_2$ have a limited influence on the uncertainty of problem $F2|p_{ij}^L \leq p_{ij} \leq p_{ij}^U|C_{max}$. Jobs from the set $\mathcal{J}^*$ may increase the uncertainty of the problem. In contrast to set $\mathcal{J}^*$, set $\mathcal{J}_0$ may be used to decrease the uncertainty of the input data in the *dynamic phase* of scheduling. In practice, it is useful to increase the cardinality of set $\mathcal{J}_0$ as much as possible. For example, one can include an additional job J_k with fixed processing times for a preventive maintenance of machine M_1 and machine M_2 into set $\mathcal{J}$. In the *dynamic*

phase of scheduling, such a job $J_k \in \mathcal{J}_0$ may be used to decrease the uncertainty of some jobs $J_i \in \mathcal{J} \setminus \mathcal{J}_0$.

6. CONCLUSION

Since the cardinality of a J-solution $S(T)$ varies for different problems $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ in the range $[1, n!]$, there is no polynomial algorithm for a direct enumeration of all permutations of a minimal dominant set $S(T)$. However, due to Theorem 1, one can construct a digraph $(\mathcal{J}, \mathcal{A}_{\preceq})$ or (if set $\mathcal{J}_0$ is empty) a digraph $(\mathcal{J}, \mathcal{A}_{\prec})$ in $O(n^2)$ time. Due to Theorems 5 and 6, digraph $(\mathcal{J}, \mathcal{A}_{\prec})$ defines a set $S(T)$ and may be considered as a condensed form of a J-solution to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$. Of course, the more job pairs involved in the binary relation $\mathcal{A}_{\preceq}$, the more redundant permutations will be deleted from set S while constructing a J-solution $S(T) \subseteq S$ to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$.

Due to Theorems 5 and 6, we can construct a J-solution $S(T)$, $|S(T)| > 1$, to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ and use it as a more general schedule form. In particular, using a J-solution, one can look for a *robust* schedule and reduce time-consuming computations (see Kouvelis et al. [2000]) by treating a minimal set $S(T)$ of dominant permutations instead of the whole set S of permutations. Of course, this reduction of computations may be essential if the cardinality of set $S(T)$ is significantly less than $|S| = n!$. If the uncertainty exceeds a certain threshold, then other approaches will outperform the approach based on Theorems 1, 2, 3, 5, 6, e.g., a fuzzy method, Slowinski and Hapke [1999], and a robust method, Kouvelis and Yu [1997]. The former method is practically useful when the scheduler has enough prior information to characterize the probability distributions of the random processing times and there is a large number of realizations of similar processes. The latter method allows us to determine the schedule with the best worst-case performance compared to the corresponding optimal schedule over all potential realizations of the job processing times.

Our method is consistent with the hierarchical approach to scheduling adopted over the last decade, and it corresponds to industrial practices, Aytug et al. [2005]. In the *static phase*, when the level of uncertainty of the input data is large, a scheduler can find a J-solution $S(T)$ to problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$. In order to find an optimal schedule, the subset of schedules from a J-solution to the subproblem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$ obtained after each decision being made in the *dynamic phase* has to keep a J-solution for the remaining subset of jobs. Matsveichuk et al. [2009] have shown by experiments that such a hierarchical approach based on a minimal dominant set $S(T)$ is efficient for problem $F2|p_{ij}^L \le p_{ij} \le p_{ij}^U|C_{max}$.

ACKNOWLEDGEMENTS

The research of the first author was supported by the World Federation of Scientists. The second and third authors were supported by Belarusian Republican Foundation for Fundamental Research (grant M08MC-027). The second author was also supported by National Science Council of Taiwan (grant 98-2811-H-002-005).

REFERENCES

A. Allahverdi, T. Aldowaisan, Yu.N. Sotskov. Two-machine flowshop scheduling problem to minimize makespan or total completion time with random and bounded setup times. *International Journal of Mathematics and Mathematical Sciences*, 39(11):2475–2486, 2003.

A. Allahverdi, Yu.N. Sotskov. Two-machine flowshop minimum-length scheduling problem with random and bounded processing times. *International Transactions in Operational Research*, 10:65–76, 2003.

H. Aytug, M.A. Lawley, K. McKay, S. Mohan, R. Uzsoy. Executing production schedules in the face of uncertainties: a review and some future directions. *European Journal of Operational Research*, 161:86–110, 2005.

O. Braun, T.-C. Lai, G. Schmidt, Yu.N. Sotskov. Stability of Johnson's schedule with respect to limited machine availability. *International Journal of Production Research*, 40(17):4381–4400, 2002.

O. Braun, N.M. Leshchenko, Yu.N. Sotskov. Optimality of Jackson's permutations with respect to limited machine availability. *International Transactions in Operational Research*, 13:59–74, 2006.

S.M. Johnson. Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly*, 1:61–68, 1954.

P. Kouvelis, R.L. Daniels, G. Vairaktarakis. Robust scheduling of a two-machine flow shop with uncertain processing times. *IIE Transactions*, 32:421–432, 2000.

P. Kouvelis, G. Yu. Robust Discrete Optimization and Its Applications. *Kluwer Academic Publishers*, Boston, The USA. 1997.

T.-C. Lai, Yu.N. Sotskov. Sequencing with uncertain numerical data for makespan minimization. *Journal of the Operational Research Society*, 50:230–243, 1999.

T.-C. Lai, Yu.N. Sotskov, N. Sotskova, F. Werner. Optimal makespan scheduling with given bounds of processing times. *Mathematical and Computer Modelling*, 26:67–86, 1997.

T.-C. Lai, Yu.N. Sotskov, N. Sotskova, F. Werner. Mean flow time minimization with given bounds of processing times. *European Journal of Operational Research*, 159: 558–573, 2004.

N.M. Leshchenko, Yu.N. Sotskov. Two-machine minimum-length shop-scheduling problems with uncertain processing times. *Proceedings of XI International Conference "Knowledge-Dialogue-Solution"*, Varna, Bulgaria, June 20-24:375–381, 2005.

N.M. Matsveichuk, Yu.N. Sotskov, N.G. Egorova, T.-C. Lai. Schedule execution for two-machine flow-shop with interval processing times. *Mathematical and Computer Modelling* 49(5-6):991–1011, 2009.

M. Pinedo. Scheduling: Theory, Algorithms, and Systems. *Prentice-Hall*, Englewood Cliffs, 1995.

R. Slowinski, M. Hapke. Scheduling Under Fuzziness. *Physica-Verlag*, Heidelberg, New York, 1999.

Yu.N. Sotskov, A. Allahverdi, T.-C. Lai. Flowshop scheduling problem to minimize total completion time with random and bounded processing times. *Journal of the Operational Research Society*, 55:277–286, 2004.

V.S. Tanaev, Yu.N. Sotskov, V.A. Strusevich. Scheduling Theory: Multi-Stage Systems. *Kluwer Academic Publishers*, Dordrecht, The Netherlands, 1994.