Production Systems and Information Engineering, Miskolc, Vol. 1. (2003), pp. 55-69

A MULTIPLE (EXTENDED) APPLICATION OF THE JOHNSON ALGORITHM FOR THE TWO-MACHINE MANUFACTURING CELL SCHEDULING BASED ON GROUP TECHNOLOGY

SÁNDOR RADELECZKI*
Institute of Mathematics, University of Miskolc
H-3515 MISKOLC, Hungary
matradi@uni-miskolc.hu

TIBOR TÓTH*
Department of Information Engineering, Institute of Information Science,
University of Miskolc
H-3515 MISKOLC, Hungary
toth@iit.uni-miskolc.hu

JÁNOS GÖNDRI-NAGY*
Department of Information Engineering, Institute of Information Science,
University of Miskolc
H-3515 MISKOLC, Hungary
iitgnagy@uni-miskolc.hu

[Received September 2002 and accepted March 2003]

* Production Information Engineering Research Team (PIERT) of the Hungarian Academy of Sciences; Department of Information Engineering of the University of Miskolc

Abstract. As is known, the *Johnson* algorithm is an exact solving method of the two-machine, one-way, no-passing scheduling tasks [1], [6], which serves as a basis for many heuristic algorithms. This paper presents the extension of Johnson's algorithm for Group Technology (GT). The task is as follows: Let us assume that in a two-machine manufacturing cell, in which two machines (A and B) of high automation and environment degree are working together in such a way that machine A is always ready to perform jobs, and machine B is working or waiting according to the timing of the work-pieces transferred from machine A to machine B; the work-pieces are arranged in groups $G_1, G_2, \ldots, G_i, \ldots, G_m$ Because of the similarities of the work-pieces in group G_i, retooling and other setups (e. g.: change of equipment) of the machines in the manufacturing cell are not necessary. Consecutive machining of the groups G_i and G_j requires retooling and other setups in the manufacturing cell. An ordering of the groups is to be determined considering all the groups so that the sum of the setup times is to be minimum for all groups.
In the paper the authors prove that the solving of this task can be traced back to the extended application of the Johnson algorithm and results an exact, closed-form optimum.
Keywords: Johnson algorithm, scheduling, Group Technology, Manufacturing Cell

1. Base Model of the Two-Machine, One-Way, No-Passing Scheduling Task

The Johnson algorithm enables the solution of sequencing tasks (here: one-way, no-passing), in which n different jobs are to be allocated to *two* consecutive workplaces (machines, equipment, $m = 2$)[7]. In certain special cases, the task can be extended also to $m = 3$. Although it cannot be used in case of $m > 3$, yet it is an important procedure, because it constitutes the basis of the heuristic methods developed for large-sized tasks. Let us consider Figure 1:

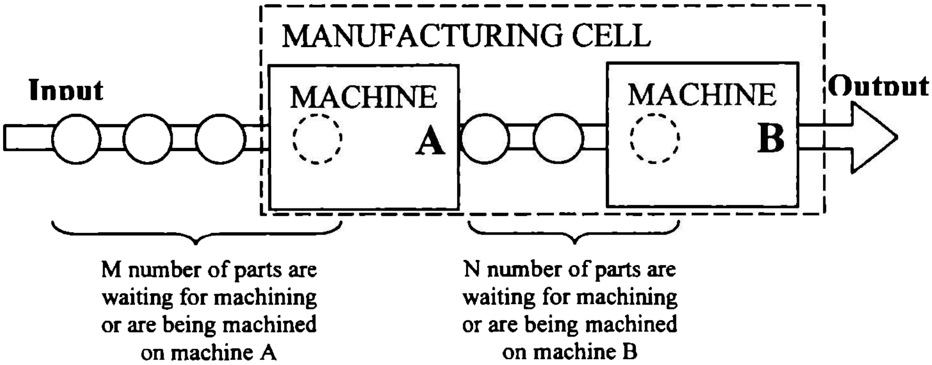

Fig. 1: Model of the two-machine manufacturing cell for deducing Johnson's algorithm

In Figure 1 a manufacturing cell consisting of two machines is demonstrated, on which the machining processes, in accordance with the machines are as follows:

$A \rightarrow$ rough boring,

$B \rightarrow$ finish boring.

It is assumed that dividing the bore machining into two phases using two machines (with different accuracy) is reasonable because of the tolerances.

The two different machining operations are to be carried out on n different parts whose characteristics are similar but their dimensions can significantly differ from one another; in addition, their arrival sequence is optional. Consequently, the throughput time for a series optional but fixed after selection can significantly differ from another variant of the same series manufactured in a different sequence.

Let us denote the time needed for machining the ith part on machine A and B with A_i and B_i, respectively. *The task is to minimize the idle time of machine B.* ("Idle time" stands for the time that elapses between the completion of job p_{i-1} and the start of job p_i.) That is, we want to determine a sequence $p_1, p_2, \ldots, p_i, \ldots, p_n$, for which the sum of idle times between finish boring parts p_j and p_{j+1} will be minimum, computing the sum for consecutive j values.

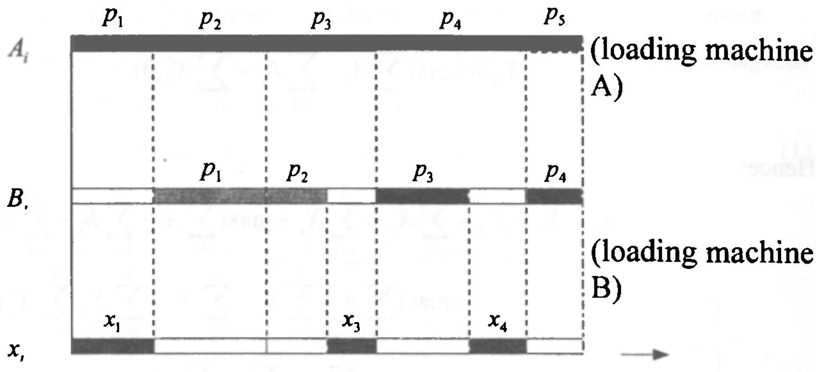

Fig. 2: Default Gantt chart for the two-machine cell model

Let us denote the time that elapses between the start of rough boring the first part and the completion of finish boring the last part by T. Let X_i be the idle time between the completion of job p_{i-1} and the start of job p_i. According to Figure 2, we can write:

$$T = \sum_{i=1}^{n} B_i + \sum_{i=1}^{n} X_i ,$$

For $\sum_i B_i$ is given and known, only $\sum_i X_i$ is to be minimized.

From Figure 2 it can be seen that:

$$X_1 = A_1, \quad X_2 = 0, \qquad \text{if } A_1 + A_2 < B_1 + X_1,$$
$$X_2 = A_1 + A_2 - B_1 - X_1, \quad \text{if } A_1 + A_2 > B_1 + X_1.$$

(The equality sign is taken into consideration in the second case, because the transition times between the machines to a first approximation are ignored.) Therefore, such an X_2 is to be determined, for which the following applies:

$$X_2 = \max(A_1 + A_2 - B_1 - X_1, 0) = \max(\sum_{i=1}^{2} A_i - \sum_{i=1}^{1} B_i - \sum_{i=1}^{1} X_i, 0).$$

Let us examine the sum $X_1 + X_2$. We can write that

$$X_1 + X_2 = X_1 + \max(A_1 + A_2 - B_1 - X_1, 0) =$$
$$= \max(A_1 + A_2 - B_1, X_1) = \max(A_1 + A_2 - B_1, A_1) =$$
$$= \max(\sum_{i=1}^{2} A_i - \sum_{i=1}^{1} B_i, A_1).$$

Similarly:

$$X_3 = \max(\sum_{i=1}^{3} A_i - \sum_{i=1}^{2} B_i - \sum_{i=1}^{2} X_i, 0).$$

Hence:

$$X_1 + X_2 + X_3 = \sum_{i=1}^{3} X_i = \sum_{i=1}^{2} X_i + \max(\sum_{i=1}^{3} A_i - \sum_{i=1}^{2} B_i - \sum_{i=1}^{2} X_i, 0) =$$
$$= \max(\sum_{i=1}^{3} A_i - \sum_{i=1}^{2} B_i - \sum_{i=1}^{2} X_i + \sum_{i=1}^{2} X_i, \sum_{i=1}^{2} X_i) =$$
$$= \max(\sum_{i=1}^{3} A_i - \sum_{i=1}^{2} B_i, \sum_{i=1}^{2} X_i) =$$
$$= \max(\sum_{i=1}^{3} A_i - \sum_{i=1}^{2} B_i, \sum_{i=1}^{2} A_i - B_1, A_1).$$

This formula can easily be extended for *n*-number idle times assuming a certain sequence $(S) = (p_1, p_2, \ldots, p_i, \ldots, p_n)$:

$$D_n(S) = \sum_{i=1}^{n} X_i = \max\left[\sum_{i=1}^{n} A_i - \sum_{i=1}^{n-1} B_i, \sum_{i=1}^{n-1} A_i - \sum_{i=1}^{n-2} B_i, \ldots, A_1\right].$$

This formula can be written in a more concise form in the following manner:

$$D_n(S) = \max_{1 \le r \le n}\left[\sum_{i=1}^{r} A_i - \sum_{i=1}^{r-1} B_i\right]$$

or in another way:

$$D_n(S) = \max_{1 \le r \le n} L_r, \quad \text{where} \quad L_r = \sum_{i=1}^{r} A_i - \sum_{i=1}^{r-1} B_i$$

Let us have some kind of series $(S^{(1)})$:

$$(S^{(1)}) = (p_1, p_2, \ldots, p_{k-1}, p_k, p_{k+1}, p_{k+2}, \ldots, p_n)$$

and a series $(S^{(2)})$, that can be obtained from $(S^{(1)})$ exchanging *k* and *k+1* with each other:

$$(S^{(2)}) = (p_1, p_2, \ldots, p_{k-1}, p_{k+1}, p_k, p_{k+2}, \ldots, p_n),$$

and let us define the sums $L_r^{(1)}$ and $L_r^{(2)}$ for the first *r* members of $(S^{(1)})$ and $(S^{(2)})$ similarly.

It is easy to see that $L_r^{(1)}$ and $L_r^{(2)}$ are the same for all $1 \le r \le n$ in the cases of series $(S^{(1)})$ and $(S^{(2)})$, except maybe the cases of $r = k$ and $r = k+1$.

Hence $D_n(S^{(1)}) = D_n(S^{(2)})$, whenever $\max(L_k^{(1)}, L_{k+1}^{(1)}) = \max(L_k^{(2)}, L_{k+1}^{(2)})$.

If $\max(L_k^{(1)}, L_{k+1}^{(1)}) \neq \max(L_k^{(2)}, L_{k+1}^{(2)})$, then one of the two series $(S^{(1)})$ and $(S^{(2)})$ is more advantageous than the other. Series $(S^{(1)})$ – in which *k+1* follows *k* – is more advantageous than series (S^2), in which *k+1 precedes k*, if

$$\max(L_k^{(1)}, L_{k+1}^{(1)}) < \max(L_k^{(2)}, L_{k+1}^{(2)}). \tag{1}$$

In detail:

$$\max(L_k^{(1)}, L_{k+1}^{(1)}) = \max(\sum_{i=1}^{k} A_i - \sum_{i=1}^{k-1} B_i, \sum_{i=1}^{k+1} A_i - \sum_{i=1}^{k} B_i),$$

$$\max(L_k^{(2)}, L_{k+1}^{(2)}) = \max\left[\sum_{i=1}^{k-1} A_i + A_{k+1} - \sum_{i=1}^{k-1} B_i, \sum_{i=1}^{k+1} A_i - \sum_{i=1}^{k-1} B_i - B_{k+1}\right]$$

Hence, we obtain:

$$\sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k+1} A_i + \max(L_k^1, L_{k+1}^1) =$$

$$= \max\left[\sum_{i=1}^{k} A_i - \sum_{i=1}^{k+1} A_i + \sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k-1} B_i, \sum_{i=1}^{k+1} A_i - \sum_{i=1}^{k+1} A_i + \sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k} B_i\right] =$$

$$= \max(-A_{k+1}, -B_k) = -\min(A_{k+1}, B_k).$$

And similarly:

$$\sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k+1} A_i + \max(L_k^2, L_{k+1}^2) =$$

$$= \max\left[\sum_{i=1}^{k-1} A_i + A_{k+1} - \sum_{i=1}^{k+1} A_i - \sum_{i=1}^{k-1} B_i + \sum_{i=1}^{k-1} B_i, \sum_{i=1}^{k+1} A_i - \sum_{i=1}^{k+1} A_i + \sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k-1} B_i - B_{k+1}\right] =$$

$$= \max\left[\sum_{i=1}^{k-1} A_i + A_{k+1} - (\sum_{i=1}^{k-1} A_i + A_k + A_{k+1}), -B_{k+1}\right] =$$

$$= \max(-A_k, -B_{k+1}) = -\min(A_k, B_{k+1}).$$

Hence, relation (1) is equivalent to the following form:

$$-\min(A_{k+1}, B_k) < -\min(A_k, B_{k+1}),$$

that is, to:

$$\min(A_{k+1}, B_k) > \min(A_k, B_{k+1})$$

Now, we can draw the conclusion that the sequence $(..., p_k, p_{k+1}, ...)$ is more advantageous than the sequence $(..., p_{k+1}, p_k, ...)$ if:

$$\min(A_k, B_{k+1}) < \min(A_{k+1}, B_k) \quad . \tag{2}$$

Let $1 \le k < l \le n$ and let us consider now the sequence $(S') = (p_1, p_2, \ldots, p_l, \ldots, p_k, \ldots, p_n)$ different from *(S)* only in the fact that in (S') the job p_l is staying in the position number *k*, and it is the job p_k in the position *k*.

The sequence *(S)* is more advantageous than (S'), if and only if

$$\min(A_k, B_l) \le \min(A_l, B_k), \quad (3)$$

which holds either $A_k \le B_l$ & $A_k \le A_l$ & $A_k \le B_k$ is true,

or $B_l \le A_k$ & $B_l \le A_l$ & $B_l \le B_k$ is true.

The first case can be expressed also in the form:

$$\min(A_k, B_k) \le \min(A_l, B_l). \quad (4)$$

Therefore, if we find a time A_k in the table of times which is less than all other A_l and B_l at the same time, then it has to be begun the sequence with p_k. If time A_k – although is one of the smallest times – is equal to another A_l or B_l, the sequence can also be begun with p_k.

The second case is equivalent to:

$$\min(A_l, B_l) \le \min(A_k, B_k). \quad (4')$$

Consequently, if we find a time B_l in the table of times, that is less than all other A_k or B_k at the same time, then the sequence to be determined must be ended with p_l. If time B_l although is one of the shortest times – is equal to another A_k or B_k, the sequence can also be ended with it.

It can be seen from the detailed deduction, that the sequence can be determined step-by-step by means of the *Johnson algorithm*. In Operations Research, the mathematical method that optimizes a series of decisions depending on one another is named *Dynamic Programming*. *Johnson's* algorithm solves actually a dynamic programming problem.

Extension of the Johnson algorithm for three machines

The *Johnson* algorithm can also be used in the following two special cases:

$$\min A_i \ge \max B_i \quad \text{or} \quad \min C_i \ge \max B_i,$$

if the three machines *A, B* and *C* with *n* jobs to be done are given. In such a case the examination of the times is executed with the sums $A_i + B_i$ and $B_i + C_i$. These "virtual machining times" are handled exactly as real machining times of two fictive machines [7].

Let us consider the following task:

The turning, milling and grinding operations are defined by periods A_i, B_i and C_i for the parts denoted by $p_1, \ldots, p_5$.

Let us start from the following table:

	Turn (A_i)	Mill (B_i)	Grind (C_i)
p_1	8	6	7
p_2	12	3	10
p_3	9	5	4
p_4	15	4	18
p_5	11	5	10

Table I

It is valid that $min\ A_i \geq max\ B_i$, because: $8 > 6$. We can compile the second table, too:

	$A_i + B_i$	$B_i + C_i$
p_1	14	13
p_2	15	13
p_3	14	9
p_4	19	22
p_5	16	15

Table II

Applying the Johnson algorithm, we get:

1.

p_1	14	13
p_2	15	13
p_4	19	22
p_5	16	15
p_3	14	9

2.

p_1	14	13
p_4	19	22
p_5	16	15
p_2	15	13
p_3	14	9

3.

p_4	19	22
p_5	16	15
p_1	14	13
p_2	15	13
p_3	14	9

The algorithm can be well followed through Table II, 1. 2. and 3.

1. We examine, which is the smallest time in the Table II; this time, it is 9 minutes.
2. If this value belongs to the *first* column of the table, then we start with the part corresponding to it.; if it belongs to the *second* column, then we end machining with the part corresponding to it.
3. We separate the corresponding row of the table, and we follow the steps 1.-2.-3. with the remaining part of the table. For the optimal scheduling sequence we get, that:

$$S = (p_4, p_5, p_1, p_2, p_3).$$

Let us take the original *ad hoc* sequence:

$$S_1 = (p_1, p_2, p_3, p_4, p_5).$$

Representing it (Figure 3):

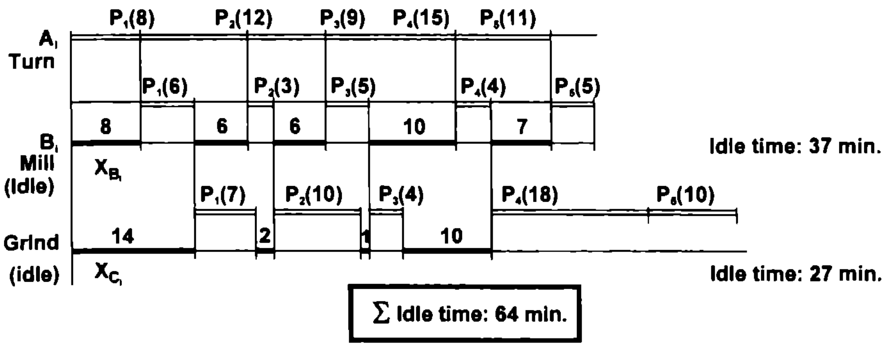

Fig. 3: Demonstrating the extension of the Johnson algorithm through a concrete example (Gantt chart, *ad hoc* sequence)

According to the *Johnson* algorithm (Figure 4):

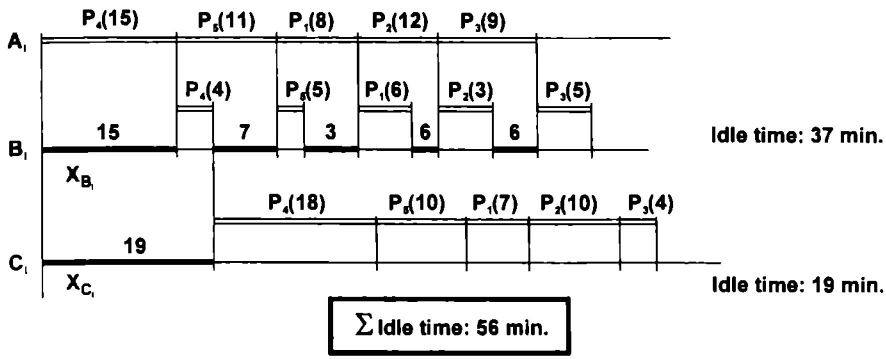

Fig. 4: Demonstrating the extension of Johnson's algorithm through a concrete example (Gantt chart, *optimal* sequence)

From the two Gantt chart it can be seen, that the Johnson algorithm decreases the idle times by 8 minutes.

2. Modification of the Base Model Considering Retooling (Setup) Times

Let $p_1, p_2, ..., p_n$ be the pieces to be machined on machine A and B, and let us denote their machining times by $A_1, A_2, ..., A_n$ and $B_1, B_2, ..., B_n$, respectively [5]. Let $x_1, x_2, ..., x_n$ denote the idle times during (or rather prior to) the machining of pieces $p_1, p_2, ..., p_n$ on machine B. Let $\pi = (i_1, i_2, \quad i_n)$ denote an arbitrary permutation of the indexes $(1, 2, \quad n)$ and let ρ_n stand for the set of all permutations of the index set $\{1, 2, ..., n\}$.

It is known, that in case of a $p_{i_1}, p_{i_2}, ..., p_{i_n}$ permutation of the pieces, idle times $x_{i_1}^{(\pi)}, x_{i_2}^{(\pi)}, ..., x_{i_n}^{(\pi)}$ that occur on machine B differ – Let us denote the sum of these with X_π.

If the pieces are ordered in the sequence $p_{j_1}, p_{j_2}, ..., p_{j_n}$ that is given by Johnson's algorithm, i.e. we consider the $\pi_J=(j_1, j_2, ..., j_n)$ permutation of the indexes $(1, 2, ..., n)$, then idle time X_{π_J} will be the minimum possible, that is

$$X_{\pi_J} = \min\{X_\pi | \pi \in \rho_n\}.$$

(1) First of all, we would like to show that if machine A does not start to operate at the $t=0$ point of time but at the point of time t (where t can also be negative, which means that machining on machine A had already been started before 0 point of time), the optimal sequence of machining – i.e. the sequence when the sum of idle times x_i on machine B is minimum – remains the one that is determined by the Johnson algorithm, that is the sequence corresponding to the index permutation π_J. If $t > 0$ then this case can also be viewed as if, the machining time of piece p_{i_1} on machine A that stands in the first position would grow from A_{i_1} to $A_{i_1}+t$, for the machining sequence $p_{i_1}, p_{i_2}, ..., p_{i_n}$ given by the permutation $\pi=(i_1, i_2, ..., i_n)$, and the machining time on machine B would remain B_{i_1}

As is known, the idle time (in case of starting at the point of time 0) for permutation π was so far given by the formula $X_\pi = \max\limits_{1\le r\le n}\left(\sum_{j=1}^{r} A_{i_j} - \sum_{j=1}^{r-1} B_{i_j}\right)$, now we get the following:

$$X'_\pi = \max_{1\le r\le n}\left(\sum_{j=1}^{r} A_{i_j} + t - \sum_{j=1}^{r-1} B_{i_j}\right) = t + \max_{1\le r\le n}\left(\sum_{j=1}^{r} A_{i_j} - \sum_{j=1}^{r-1} B_{i_j}\right) = X_\pi + t$$

Now:

$$\min\left(X'_\pi | \pi \in \rho_n\right) = \min\left(t + X_\pi | \pi \in \rho_n\right) = t + \min\left(X_\pi | \pi \in \rho_n\right).$$

Since the minimum involved in the latter sum turns up just in the permutation relevant to the Johnson algorithm, therefore the idle time X'_π will also be minimum, if $\pi = \pi_J$; *i.e. if the pieces are further ordered according to the Johnson algorithm*. Moreover, the idle time that results in this manner is exactly $t + X_{\pi_J}$ – i.e. it is longer exactly by t time units (than in case of starting at *0* point of time).

If $t<0$, then this situation can also be seized as if we started measuring the time earlier than at *0*, but at t point of time; yet machine *B* would come into operation only with $|t|$-time later.

And this latter case can be considered, as if the machining time of the first one, part p_{i_1} on machine *B* increased from B_{i_1} to $B_{i_1} + |t| = B_{i_1}$ for the machining sequence concerning the permutation of the indexes $\pi=(i_1, i_2, ..., i_n)$, and the machining time on *A* did not alter.

Now $\sum_{j=1}^{r-1} B'_{i_j} = B_{i_1} - t + \sum_{j=2}^{r-1} B_{i_j} = \sum_{j=1}^{r-1} B_{i_j} - t$, thus now the total idle time X'_π is as follows:

$$X'_\pi = \max\left(\max_{1\le r\le n}\left(\sum_{j=1}^{r} A_{i_j} - \sum_{j=1}^{r-1} B_{i_j} + t\right);0\right) = \max\left(t + \max_{1\le r\le n}\left(\sum_{j=1}^{r} A_{i_j} - \sum_{j=1}^{r-1} B_{i_j}\right);0\right) = \max(t + X_\pi;0)$$

– considering the fact that the idle time X'_π (for any $t<0$) can only be positive or *0*. Since for the index-permutation π_j determined by the Johnson-algorithm it is true that $X_{\pi_J} \le X_\pi$ for all $\pi \in \rho_n$, therefore $\max(t + X_{\pi_J};0) \le \max(t + X_\pi;0)$, i.e. the Johnson algorithm gives still the least idle time possible, which idle time is $t + X_{\pi_J}$ if this latter number is positive, and *zero* if $t + X_\pi < 0$.

Summing up the $t>0$ and the $t<0$ cases we can say that the Johnson algorithm derives in both cases such a sequence of the pieces, beside which the total idle time is the minimum possible and both cases fit the relation that is follows: $X'_{\pi_J} = \max(X_{\pi_J} + t;0)$.

(2) As an application of the above-mentioned ones, let us consider the situation in which the machining process on machine *A* and machine *B* is preceded by the setup times S_A and S_B. This case can also be viewed as if machine *A* in comparison with machine *B* were available only in the point of time S_A-S_B. According to the above model, now the time shift on machine *A* is $t=S_A-S_B$. To this, according to item (1), the optimal sequence of pieces remains the one that is determined by the Johnson algorithm; denoting the given optimal total times with X or in case of considering setup times with X', the relation between them is as follows:

$$X' = \max(X + t;0) = \max(X + S_A - S_B;0).$$

(Note: Here the setup time S_B of machine *B* is not considered an idle time!)

3. Extension of the Modified Base Model for Group Technology

Hereafter, let us make an attempt at an optimal fitting of piece groups [3], [4] by means of tracing back it to Johnson's algorithm.

It is assumed that it is in groups G_1, G_2, G_m advantageous for the pieces to be machined (on machine A and B) because of some economic requirements, where the setup times S_{A_i} and S_{B_i} on machine A and B belong to each group G_i (that is the setup time of the machines to be prepared for the machining of group G_i). If the parts $p_{i_1}, p_{i_2},..., p_{i_{N_i}}$ belong to group G_i, then let us denote their times needed to be machined on machine A and B with $A_{i_1}, A_{i_2},..., A_{i_{N_i}}$, and with $B_{i_1}, B_{i_2},..., B_{i_{N_i}}$

Moreover, let us introduce the notations $A_i = S_{A_i} + \sum_{j=1}^{N_i} A_{i_j}$ and $B_i = S_{B_i} + \sum_{j=1}^{N_i} B_{i_j}$. According to item (2), the total idle time for group G_i will be minimum, if the pieces i.e. the corresponding number pairs $(A_{i_1}, B_{i_1}), (A_{i_2}, B_{i_2}),..., (A_{i_{N_i}}, B_{i_{N_i}})$ are ordered in accordance with the Johnson algorithm. Let us denote the so arisen total idle time (at which the setup times have not been considered yet) for group G_i *with* X_i. (i.e. $X_i = \sum_{j=1}^{N_i} x_{i_j}$, if the pieces are ordered in the group according to the Johnson algorithm).

Let us have a look at the total idle time $X_i^{'}$ that is arisen during the machining of the pieces of group G_i, if the group machining is executed in the sequence of G_1, G_2, G_{i-1}, G_i, G_m.

The total idle time for group G_i will namely be influenced by the fact which point of times $T_{A_{i-1}}, T_{B_{i-1}}$ machine A after completing the pieces of the groups G_1, G_2, G_{i-1}, and machine B after completing the pieces of the groups G_1, G_2, G_{i-1} are available at.

Accurately, the reason for the idle time X_i to be altered is that machine A in comparison to machine B is available with a time shift of $t_{i-1} = T_{A_{i-1}} - T_{B_{i-1}}$ Thus, the new idle time $X_i^{'}$ is as follows: $X_i^{'} = \max(X_i + t_{i-1}, 0) = \max(X_i^{'} + T_{A_{i-1}} - T_{B_{i-1}}, 0)$. The time $T_{A_{i-1}}$ means evidently the sum $A_1+A_2+...+A_{i-1}$ of the times (where $A_k = S_{A_k} + \sum_{j=1}^{N_k} A_{k_j}$ – so it contains the setup times $S_{A_1}, S_{A_2},..., S_{A_{i-1}}$ too.), whilst the time $T_{B_{i-1}}$ is the sum of the times $B_1 + X_1^{'} + B_2 + X_2^{'} + ... + B_{i-1} + X_{i-1}^{'}$ (where: $B_k = S_{B_k} + \sum_{j=1}^{N_k} B_{k_j}$), so

$$t_{i-1}=\sum_{k=1}^{i-1}A_k-\sum_{k=1}^{i-1}B_k-\sum_{k=1}^{i-1}X_k' .$$

Let us observe that in case of group G_1, which is preceded by no other groups but the setup times S_{A_1} and S_{B_1}, $T_{A_0}=S_{A_1}$, $T_{B_0}=S_{B_1}$, thus it can be said that $t_0=S_{A_1}-S_{B_1}$, and for X_1' we get that $X_1'=\max(X_1+t_0;\,0)=\max(X_1+S_{A1}-S_{B1};\,0)$.

Similarly, $X_2'=\max(X_2+t_1;\,0)=\max(X_2+A_1-B_1-X_1';0)$.

Thus $\quad X_1'+X_2'=\max(X_2+A_1-B_1;\,X_1')=\max(X_2+A_1-B_1,X_1+S_{A_1}-S_{B_1};\;0)$

$$X_3'=\max(X_2+t_2;\;0)=\max(X_3+(A_1+A_2)-(B_1+B_2)-(X_1'+X_2');\;0)$$

Thus

$$X_1'+X_2'+X_3'=\max(X_3+(A_1+A_2)-(B_1+B_2);\,X_1'+X_2')=$$
$$+\max(X_3+(A_1+A_2)-(B_1+B_2);\,X_2+A_1-B_1;X_1+S_{A_1}-S_{B_1};0).$$

In general

$$X_i'=\max(X_i+t_{i-1},0)=\max\left(X_i+\sum_{k=1}^{i-1}A_k-\sum_{k=1}^{i-1}B_k-\sum_{k=1}^{i-1}X_k';\;0\right)$$

and

$$X_1'+X_2'+...,+X_i'=\max\left(X_i+\sum_{k=1}^{i-1}A_k-\sum_{k=1}^{i-1}B_k;\,\sum_{k=1}^{i-1}X_k'\right),$$

therefore for the total idle time $X=X_1'+X_2'+...+X_m'$ the following formula is obtained:

$$X=\max\left(X_m+\sum_{k=1}^{m-1}A_k-\sum_{k=1}^{m-1}B_k;\;X_{m-1}+\sum_{k=1}^{m-2}A_k-\sum_{k=1}^{m-2}B_k;...;X_2+A_1-B_1;\;X_1+S_{A_1}-S_{B_1};\,0\right)$$

that can be written in a more concise manner as is follows:

$$X=\max_{0\le r\le m}(T_r)\text{, where } T_0=0 \text{ and } T_r=X_r+\sum_{i=1}^{r-1}A_i-\sum_{i=1}^{r-1}B_i$$

(Note: A_0 is regarded as S_{A_1}, and B_0 as S_{B_1}).

Let us consider now the group sequences

S_1: $G_1,G_2,...,G_k,...,G_l,...,G_m$ and

S_2: $G_1,G_2,...,G_l,...,G_k,...,G_m$

that differ from each other only in the fact that in the second series it is the group G_l staying in the position number k, and it is the group G_k. in the position number l.

Since in this case the computations can be fulfilled in a very analogous way as in the case when k and l are consecutive numbers, for the sake of clarity we will confine ourselves only to the latter case.

Let the series of groups be the followings:

S_1: $G_1, G_2, \ldots, G_{k-1}, G_k, G_{k+1}, \ldots, G_m$ and

S_2: $G_1, G_2, \ldots, G_{k-1}, G_{k+1}, G_k, \ldots, G_m$,

Let us denote the part sums T_r that belong to the first series of groups with T_r^1, and the ones that belong to the second series of groups with T_r^2

Obviously, the machining sequence that is given by series S_1 is more efficient than the machining sequence corresponding to S_2 *if and only if*

$$\max\left(T_k^1, T_{k+1}^1\right) < \max\left(T_k^2, T_{k+1}^2\right). \qquad (*)$$

Now: $$\max\left(T_k^1, T_{k+1}^1\right) = \max\left(X_k + \sum_{i=1}^{k-1} A_i - \sum_{i=1}^{k-1} B_i;\ X_{k+1} + \sum_{i=1}^{k} A_i - \sum_{i=1}^{k} B_i\right),$$

thus: $$\sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k-1} A_i + \max\left(T_k^1, T_{k+1}^1\right) = \max(X_k;\ X_{k+1} + A_k - B_k).$$

On the other hand:

$$\max\left(T_k^2, T_{k+1}^2\right) = \max\left(X_{k+1} + \sum_{i=1}^{k-1} A_i - \sum_{i=1}^{k-1} B_i;\ X_k + \sum_{i=1}^{k-1} A_i + A_{k+1} - \sum_{i=1}^{k-1} B_i - B_{k+1}\right),$$

– taking into consideration that the total idle time during the machining of group G_{k+1} is X_{k+1}, and during the machining of group G_k is X_k.

Thus

$$\sum_{i=1}^{k-1} B_i - \sum_{i=1}^{k-1} A_i + \max\left(T_k^2, T_{k+1}^2\right) = \max\left(X_{k+1};\ X_k + A_{k+1} - B_{k+1}\right).$$

According to the researches, the inequality (*) is equivalent to the inequality as follows:

$$\max(X_k;\ X_{k+1} + A_k - B_k) < \max(X_{k+1};\ X_k + A_{k+1} - B_{k+1})$$

Adding $-X_k - X_{k+1}$ to both side of the inequality, we get

$$\max\left(-X_{k+1};\ A_k - B_k - X_k\right) < \max\left(-X_k;\ A_{k+1} - B_{k+1} - X_{k+1}\right), \text{ i.e.}$$

$$\max\left(-X_{k+1};\ -\left(B_k + X_k - A_k\right)\right) < \max\left(-X_k;\ -\left(B_{k+1} + X_{k+1} - A_{k+1}\right)\right).$$

Since $X_i \geq 0$ for all $i=1,2,\ldots,n$ and since $B_i + X_i - A_i \geq 0$ is true, all four numbers in parentheses are negative or 0, we can write:

$$-\min\left(X_{k+1}; B_k + X_k - A_k\right) < -\min\left(X_k; B_{k+1} + X_{k+1} - A_{k+1}\right),$$

wherefrom we get that (*) is equivalent to the inequality

$$\min\left(X_{k+1};\ B_k + X_k - A_k\right) > \min\left(X_k;\ B_{k+1} + X_{k+1} - A_{k+1}\right)$$

therefore in this case serial S_1 means a more advantageous sequence than series S_2.

Now returning to the general case when

S_1: $G_1, G_2, \ldots, G_k, \ldots, G_l, \ldots, G_m$ and

S_2: $G_1, G_2, \ldots, G_l, \ldots, G_k, \ldots, G_m$ (where $k<l$),

we would get in analogous way that S_l is more advantageous than sequence S_2 if and only if

$$\min(X_l, B_k + X_k - A_k) \ge \min(X_k, B_l + X_l - A_l),$$

namely, if

$$\min(X_k;\ B_l + X_l - A_l) \le \min(X_l;\ B_k + X_k - A_k).$$

Apparently, this same criterion applies to such an abstract Johnson-sequence that consists of the following number pairs:

$$(X_1;\ B_1 + X_1 - A_1), \ldots, (X_k;\ B_k + X_k - A_k), \ldots, (X_m;\ B_m + X_m - A_m).$$

Now in order to trace back the optimal sequencing problem of the groups $G_1, G_2, \ldots, G_k, \ldots, G_m$ to the sequencing problem of these number pairs according to the Johnson algorithm, it is sufficient to show that were such fictive parts $F_1, \ldots, F_k, \ldots, F_m$ considered instead of the groups $G_1, \ldots, G_k, \ldots, G_m$, whose work time on machine A are $X_1, \ldots, X_k, \ldots, X_m$, and whose work time counted for machine B are $B_1 + X_1 - A_1, \ldots, B_k + X_k - A_k, \ldots, B_m + X_m - A_m$, then we would obtain the same idle time and the same optimal sequence as in case of the machining of the piece groups $G_1, \ldots, G_k, \ldots, G_m$.

Let us denote the idle times for the fictive serial with $X_1^{"}, \ldots, X_k^{"}, \ldots, X_m^{"}$, and their sums with $X^{"}$

According to the theory of Johnson's algorithm:

$$X^{"} = \sum_{i=1}^{m} X_i^{"} = \max_{1\le r\le m} L_r \text{, where } L_r = \sum_{i=1}^{r} X_i - \sum_{i=1}^{r-1}\big((B_i + X_i) - A_i\big) = X_r + \sum_{i=1}^{r-1} A_i - \sum_{i=1}^{r-1} B_i,$$

hence $$X^{"} = \max_{1\le r\le m}\left(X_r + \sum_{i=1}^{n-1} A_i - \sum_{i=1}^{n-1} B_i \right) = \sum_{i=1}^{m} X_i^{'} = X^{'}$$

Because as for an optional permutation $\pi = (i_1, i_2, \ldots, i_n)$ of the indexes the total idle times of the group series $G_1, G_2, \ldots, G_k, \ldots, G_m$ and the fictive series of pieces $F_1, F_2, \ldots, F_r, \ldots, F_m$ are the same, i.e. $X_\pi^{'} = X_\pi^{"}$, therefore it will be the same permutation $i_1, i_2, \ldots, i_m$ of the indexes that will provide the smallest total idle time for the series $G_1, G_2, \ldots, G_k, \ldots, G_m$, as for the abstract series $F_1, F_2, \ldots, F_l, \ldots, F_m$; and this is exactly the same that we have obtained for the latter series by means of the Johnson algorithm.

Therefore, first and last, it can be said that the group series G_1, G_2, ..., G_m can truly be substituted for the fictive piece series $F_1, F_2, \ldots, F_m$, where the machining time of an F_k "piece" on machine A is treated as X_k, and its machining time on machine B is treated as $B_k + X_k - A_k$.

Purely and simply, these quantities can be explicated as:

$$X_k = \sum_{j=1}^{N_k} x_{k_j} \text{ and } B_k + X_k - A_k = S_{B_k} + \sum_{j=1}^{N_k} B_{k_j} + \sum_{j=1}^{N_k} X_{k_j} - \left(S_{A_k} + \sum_{j=1}^{N_k} A_{k_j} \right).$$

Acknowledgements: The work this paper is based on has been supported by the OTKA (Hungarian Scientific Research Fund) project entitled ***„Application of Concept Lattices and Fuzzy Methods in Group Technology"*** (No.: T030243, headed by Tóth, T.). In addition, a part of this research work has been done within the framework of ***Production Information Engineering Research Team*** (PIERT) established at the Department of Information Engineering by the Hungarian Academy of Sciences in 1999. The financial support of the research by the two sources is gratefully acknowledged. Moreover, the support provided by the OTKA (Grant No. T034137) is appreciatively acknowledged.

REFERENCES

1. BRUCKER, P.: *Scheduling Algorithms,* Springer-Verlag Berlin, Heidelberg, 1998.
2. CORMEN, T. H.; LEISERSON, C E.; RIVEST, R. L.: *Introduction to Algorithms,* MIT Press, London, 2001.
3. KAMRANI, A. K.; LOGENDRAN, R.: *Group Technology and Cellular Manufacturing: Methodologies and Applications,* Gordon and Breach Science Publishers, Amsterdam, 1998.
4. NC/CIM GUIDEBOOK (1998), MODERN MACHINE SHOP: *Group Technology and Manufacturing Cells*, pp. 230-242.
5. RADELECZKI, S., and TÓTH, T.: *Connection of Concept Lattices and Fuzzy Methods and Their Application in Group Technology* (in Hungarian), Research Report. University of Miskolc, 1999.
6. ROSEN, K. H.: *Discrete Mathematics and Its Applications,* McGraw-Hill, New York, 1988.
7. TÓTH, T.: *Design and Planning Principles, Models and Methods in Computer Integrated Manufacturing* (in Hungarian), Publisher of the University of Miskolc, 1998.