PERIODICA POLYTECHNICA SER. CIVIL ENG. VOL. 37, NO. 3, PP. 249–260 (1993)

A DESCRIPTIVE PROOF FOR JOHNSON'S ALGORITHM FOR SOLVING TWO-MACHINE FLOW-SHOP PROBLEM

Zoltán A. VATTAI

Department of Building Management
Technical University of Budapest
H–1521 Budapest, Hungary
Tel: (36 1) 1813–377
FaX: (36 1) 1666–808

Received: June 2, 1993

Abstract

This paper shows a demonstrative proof of Johnson's algorithm to the problem $F2||C^{\max}$. By the help of the applied method an algorithm can be formed to solve $F2|$ overlap $|C^{\max}$ problem too.

Keywords: scheduling, sequencing, flow-shop, operations research.

1. Introduction

At the Department of Building Management of TUB researches have been performed for years aiming to solve $F|$ overlap $|C^{\max}$ problem typical in construction industry. During these investigations computer routines have been developed to control hypotheses and theories. While mathematically founding the theory of the applied methods, a descriptive by-product was discovered: a demonstrative proof of Johnson's algorithm for problem $F2||C^{\max}$. This paper deals with that interesting detail.

The problem can be summarized as follows:

There are m jobs to be performed by two parallel machines. Production times on both machines are known for each job ($T_{i,\ell}$). Both machines should collaborate in executing each job and the order of machines should be the same for each job too (*two-stage flow-shop problem*). Machines should perform jobs in the same order (*no passing allowed*). The second machine can start work on a job after the first one finished that (*no overlap allowed*). A machine should work on a job with no break until completion (*preemption not allowed*). The aim is to determine the order of jobs for execution in order to get the shortest overall execution time. (No release times, due dates or precedences should be considered.) Borrowing the code system offered by Graham, Lenstra and others in 1981 [NATO-ASRI, 1981] we denote this problem as $F2||C^{\max}$. To solve this problem, Johnson gave a heuristic algorithm in 1954 forming a milestone in research of flow-shop problems. A lot of attempts to solve multi-stage flow-shop problems have originated from his solution (JOHNSON, 1954; SZWARC, 1978). In this paper we add a surplus condition to the problem: both machines should work without any break (*no idle times allowed*). At the end of the discussion we show that this condition has no effect on the solution but will help us to demonstrate the correctness of Johnson's algorithm.

We shall use the following denotations:

$T_{i,1}$ and $T_{i,2}$: production times on job i on the first and on the second machine (input of the solution),

T : overall execution time ($C^{\max}$).

The others will be described in text as necessary.

Johnson's algorithm:

Allocate the jobs from the first and from the last position of the schedule considering them in ascending order of production times:

1 – If the considered production time occurs on the first machine, allocate the job to the start of schedule (after the already scheduled ones).

2 – If the considered production time occurs on the second machine, allocate the job to the end of schedule (before the already scheduled ones).

3 – If the production times on considered job are the same on both machines, the decision is up to you (whether to perform 1 or 2).

For demonstration while proofing we shall use two-dimensional plotting of schedules, a kind of progression curve system frequently used in construction management (PILCHER, 1976) (see *Fig. 1*).

Additional denotations:

CR_i: minimal succession time between the two machines on job i that should elapse both at the start and finish
It is borrowed from network techniques, precedence diagramming; read as : *CRitical succession time* (MODER et al., 1983).

S_i: minimal succession time between the two machines on job i at the start

F_i: minimal succession time between the two machines on job i at the finish

Steps of proof:

Concentrating on succession times, in the second chapter we shall define a *quasi O-shaped* schedule and we show that there must be an optimal schedule of that kind (Lemma 1).

In chapter 3 we define an *absolute O-shaped* schedule and we prove the existence of (at least one) optimal solution of that kind (Lemma 2).

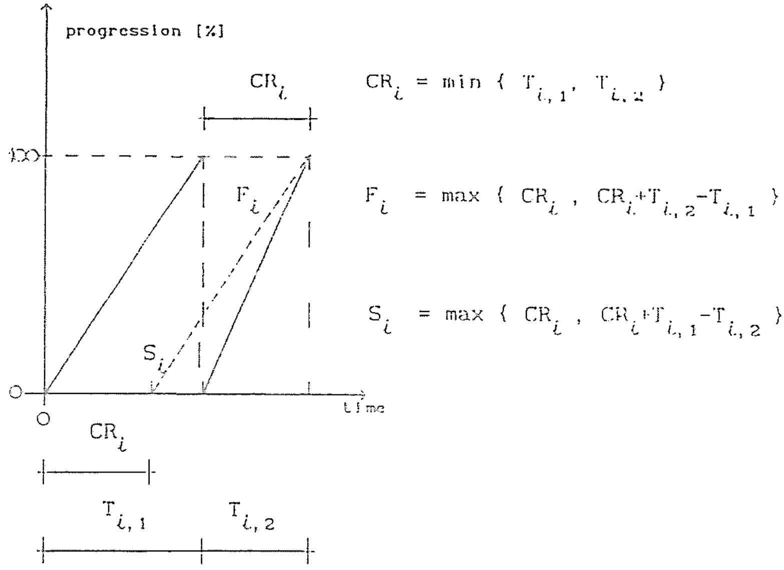


Fig. 1. Individual (minimal) schedule of job i

In the final chapter we complete the proof stating the conclusion: Johnson's algorithm results in an *absolute O-shaped* schedule (Theorem 1 and Conclusion).

2. Quasi O-shaped Schedule

DEFINITION: A schedule is called *quasi O-shaped* if no pair of jobs can be found in that for which

$$F_i < S_i \text{ and } F_j > S_j \mid i < j. \tag{R.1}$$

LEMMA 1. *There exists an optimal schedule, which is quasi O-shaped!*

PROOF: Let us assume that we found an optimal schedule that is not quasi O-shaped. In this case we surely find a pair of jobs which satisfies (R.1). If more, consider the pair (i and j) with jobs the closest to each other ($j - i = \min$).

In this case there can only be jobs if any between i and j for those

$$F_k = S_k \mid i < k < j,$$

but their presence has no effect on the logic of proof. So, practically, we assume that $j = i + 1$.

We have to examine two situations according to the succession times at joining sub-schedules.

I. The succession time at joining the predecessor sub-schedule is larger than or equal to the succession time at joining the successor sub-schedule ($D_p \geq D_s$) (see *Fig. 2a*):

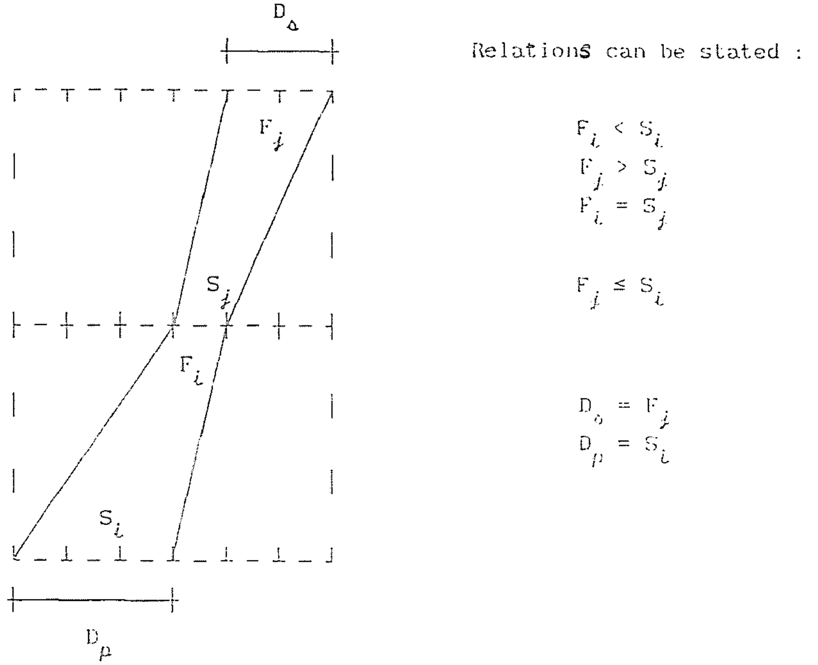


Fig. 2a. Schedule of job i and that of job j in an optimal but *non quasi O-shaped* solution (case I)

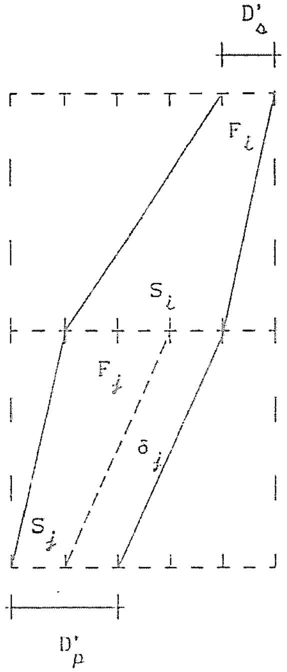
Let us modify the schedule with interchanging jobs i and j (*Fig. 2b*). After this, the following can be stated:

$$D'_s = F_i = S_j < F_j = D_s$$

and

$$D'_p = S_j + \delta_j = S_j + S_i - F_j < S_i = D_p.$$

II. The succession time at joining the predecessor sub-schedule is less than the succession time at joining the successor sub-schedule ($D_p < D_s$) (*Fig. 2c*):



Relations can be stated :

$$F_i < S_i$$
$$F_j > S_j$$
$$F_i = S_j$$

$$\delta_j = S_i - F_j$$

$$D'_s = F_i$$
$$D'_p = S_j + \delta_j$$

Fig. 2b. Forming *quasi O-shaped* schedule (case I)

Let us modify the schedule with interchanging jobs i and j (*Fig. 2d*). After this, the following can be stated:

$$D'_p = S_j = F_i < S_i = D_p \qquad and$$
$$D'_s = F_i + \delta_i = F_i + F_j - S_i < F_j = D_s.$$

As it can be seen after modifications the succession times at joining predecessor and successor sub-schedules had increased neither in case I nor in case II. In this way we can form quasi O-shaped schedule originating from any other.

From the point of view of this proof it had no importance whether in schedule of job i or in that of job j there were waiting times between the two machines in the original solution or not.

3. Absolute O-shaped schedule

DEFINITION: Let us denote with g the last job in a quasi O-shaped schedule for which $S_i < F_i$, and with h we denote the first job in the same schedule

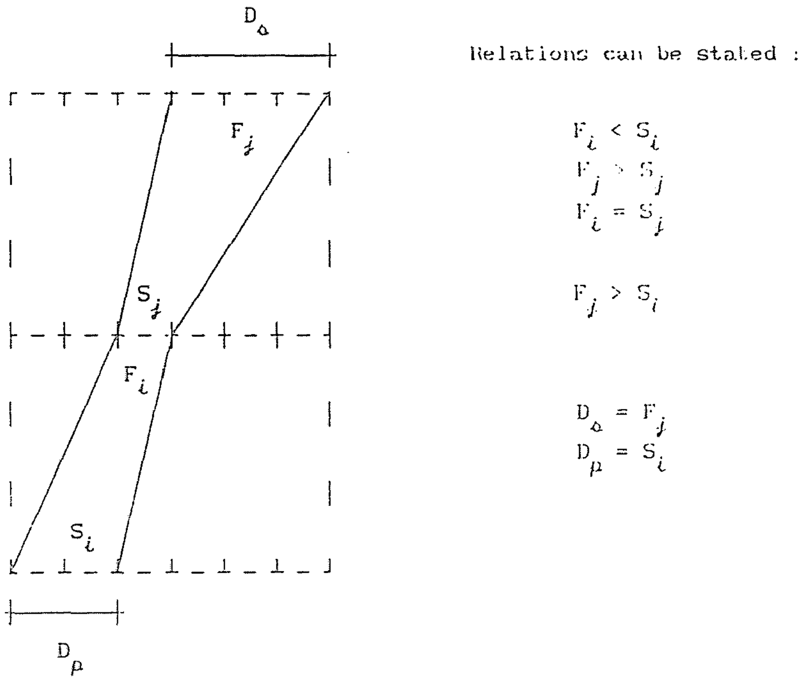


Fig. 2c. Schedule of job i and that of job j in an optimal but *non quasi O-shaped* solution (case II)

for which $S_i > F_i$. A quasi O-shaped schedule is called *absolute O-shaped* if there can't be found any pair of jobs in that for which

$$S_i > S_j \mid i < j \leq g \tag{R.2}$$

or

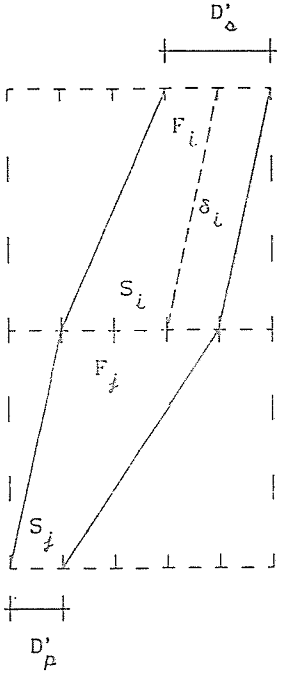
$$F_i < F_j \mid h \geq i < j. \tag{R.3}$$

LEMMA 2. *There exists an optimal schedule, which is absolute O-shaped!*

PROOF: Let us assume that we found a quasi O-shaped optimal schedule that is not absolute O-shaped. In this case we surely find a pair of jobs which satisfies (R.2) or (R.3). Let us assume that we found a pair satisfying (R.3). If more, consider the pair (i and j) with jobs the closest to each other ($j - i = \min$).

In this case there can only be jobs if any between i and j for those

$$F_k = S_k \qquad \mid \qquad i < k < j,$$



Relations can be stated :

$F_i < S_i$
$F_j > S_j$
$F_i = S_j$

$\delta_i = F_j - S_i$

$D'_s = F_i + \delta_i$
$D'_p = S_j$

Fig. 2d. Forming *quasi O-shaped* schedule (case II)

but their presence has no effect on the logic of proof. So, practically, we assume that $j = i + 1$.

We have to examine two situations according to succession times at joining sub-schedules in schedule of job i and in that of job j.

I. Succession time at finishing job j is larger than or equal to minimal succession time at starting job i ($F_j \geq S_i$) (*Fig. 3a*):
Let us modify the schedule with interchanging jobs i and j (*Fig. 3b*). After this the following can be stated:

$$D'_s = F_i + \delta'_i = F_i + F_j - S_i \leq F_j = D_s \qquad \textit{and}$$
$$D'_p = S_j = F_i = \delta_i = F_i + S_j - F_i \leq S_i + S_j - F_i = D_p.$$

II. Succession time at finishing job j is less than minimal succession time at starting job i ($F_j < S_i$) (*Fig. 3c*):
Let us modify the schedule with interchanging jobs i and j (*Fig. 3d*). After this the following can be stated:

$$D'_s = F_i < F_j = D_s \qquad \textit{and}$$
$$D'_p = S_j + \delta_j = S_j + S_i - F_j < S_j + S_i - F_i = D_p.$$

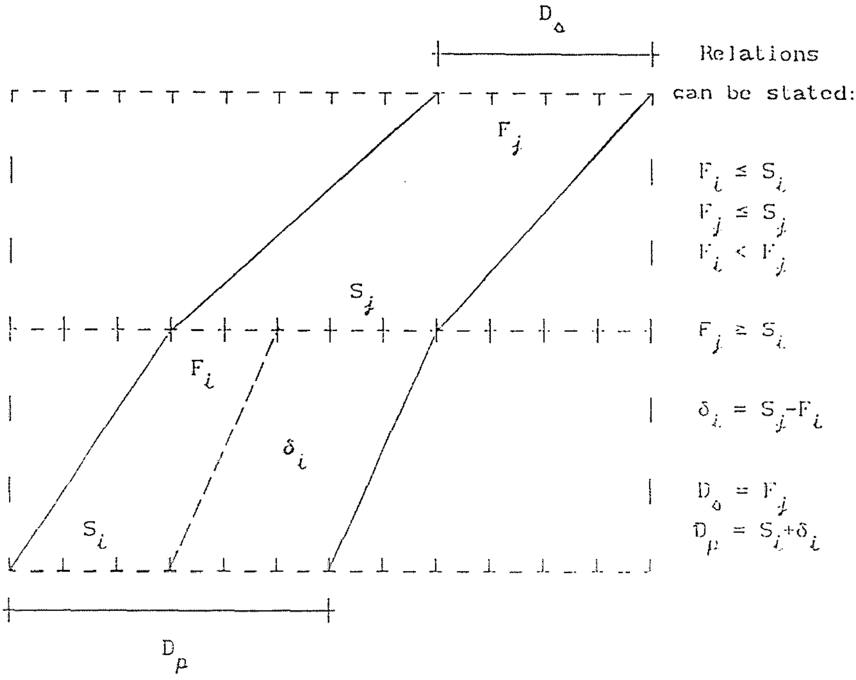


Fig. 3a. Schedule of job i and that of job j in an optimal *quasi-* but not *absolute O-shaped* solution (case I)

As it can be seen, after modifications the succession times at joining predecessor and successor sub-schedules had increased neither in case I nor in case II.

From the point of view of this proof it had no importance whether in schedule of job j there was waiting time between the two machines in the original solution or not.

Similar proof can be applied in the case of finding a pair of jobs satisfying (R.2).

In this way we can form absolute O-shaped schedule originating from any quasi O-shaped one.

4. Optimality

THEOREM 1. *If a schedule is found to be absolute O-shaped, we can be sure that it is an optimal solution of our problem.*

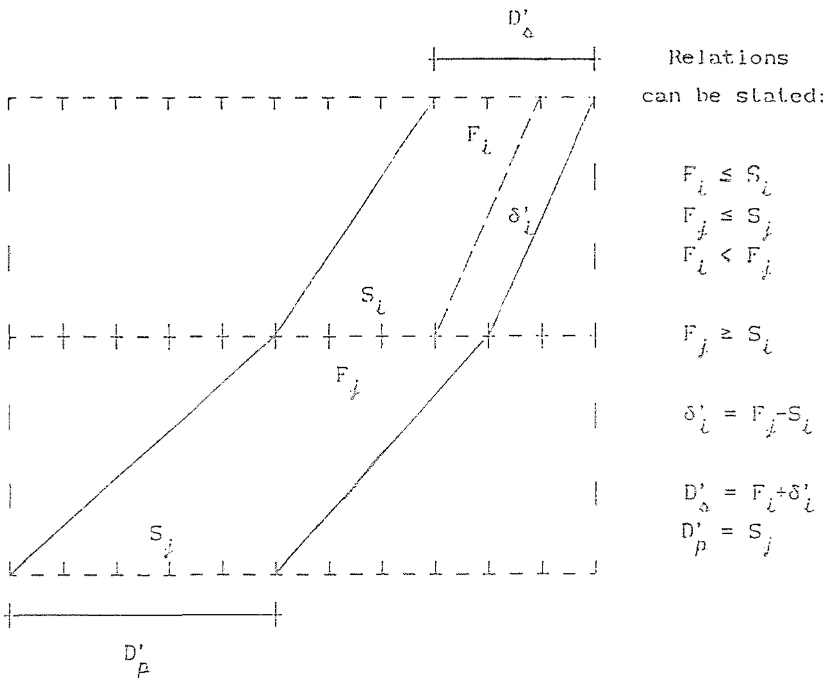


Fig. 3b. Forming *absolute O-shaped* schedule (case I)

PROOF: The total execution time of any schedule (T) can be divided into two parts:

- The sum of production times on the first machine (T'_f)
- and the succession time between the two machines at finishing the last job (T''_f),

or

- The sum of production times on the second machine (T'_s)
- and the succession time between the two machines at starting the first job (T''_s) (*Fig. 4*).

Considering that $\sum_{i=1}^{m} T_{i,1} = T'_f$ and $\sum_{i=1}^{m} T_{i,2} = T'_s$ are constants in any schedule, $\min(C^{\max}) = \min(T)$ occurs when $T''_s = T''^{\min}_s$ or $T''_f = T''^{\min}_f$. Nevertheless it is true for any absolute O-shaped schedule — by definition.

CONCLUSION: Johnson's algorithm for problem $F2||C^{\max}$ results in an absolute O-shaped schedule, consequently, it serves optimal solution.

As it was mentioned in the first chapter (Introduction), we handled the problem with a surplus condition of no idle times allowed. Finally, we

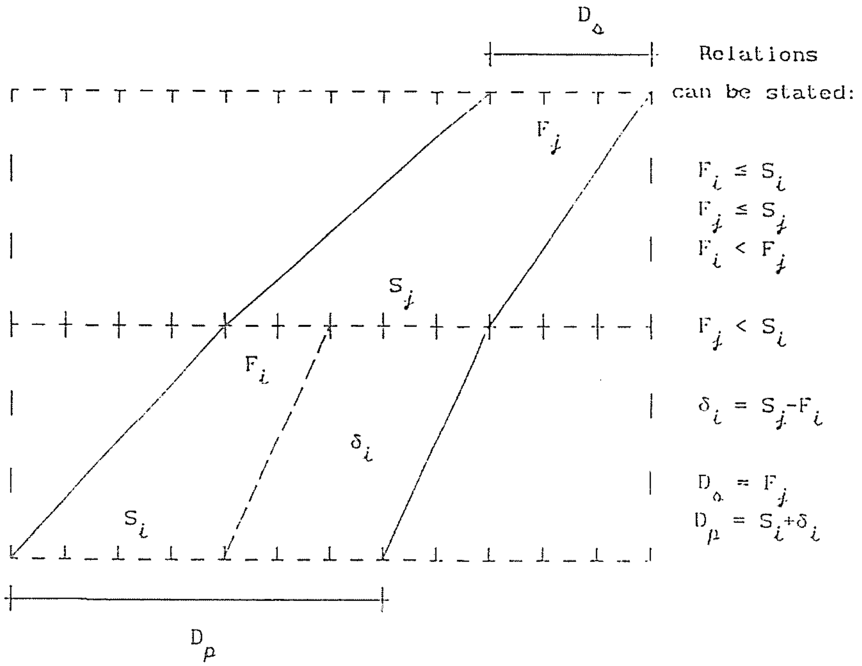


Fig. 3c. Schedule of job i and that of job j in an optimal *quasi-* but not *absolute O-shaped* solution (case II)

show that this restriction in the case of two machines has no effect on the problem.

THEOREM 2. *The sequence of jobs in an optimal solution of the problem* $F2|\,\mathrm{idle}\,|C^{\max}$ *is the optimum for problem* $F2||C^{\max}$, *too.*

PROOF: In the optimal solution of problem $F2|\,\mathrm{idle}\,|C^{\max}$, there may be idle times in the schedule of the first or of the second machine. In this case moving the schedules of the first machine toward the start or moving those of the second machine toward the finish, the idle times can be eliminated without disturbing the target value (*Fig. 5*).

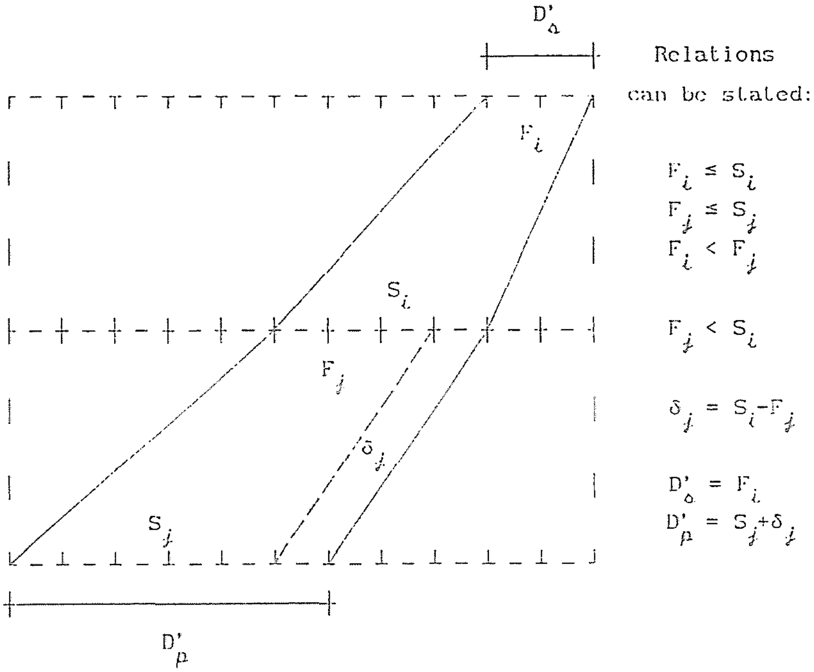


Fig. 3d. Forming *absolute O-shaped* schedule (case II)

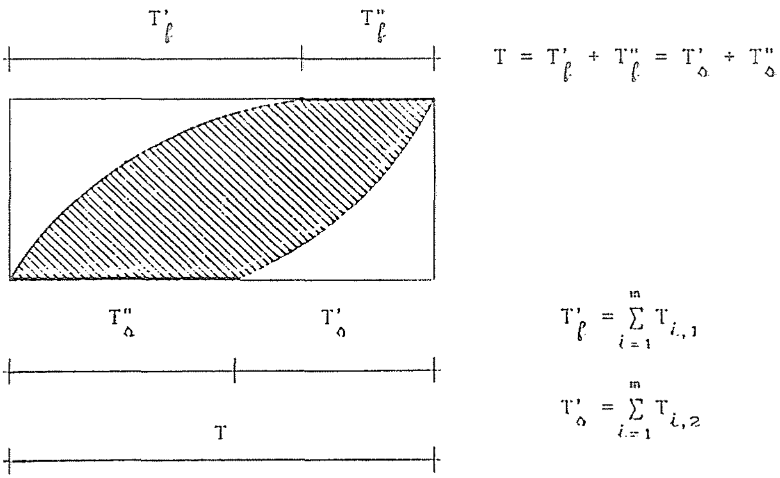


Fig. 4. Optimality of a schedule depends on the values of T''_f and T''_s

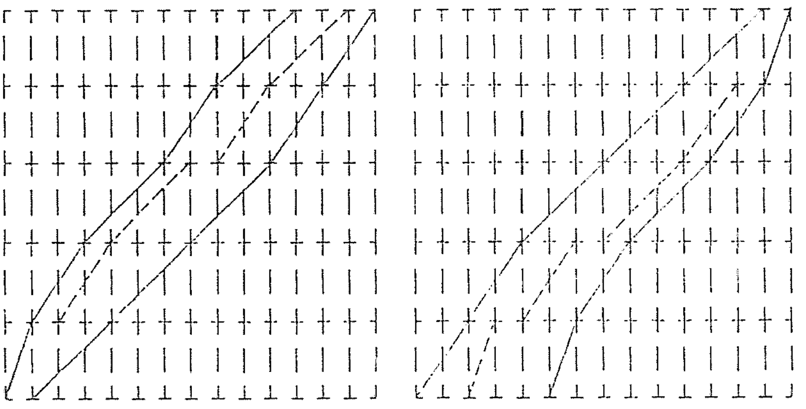


Fig. 5. Transforming optimal solution of problem $F2|\,\mathrm{idle}\,|C^{\max}$ to an optimal solution of problem $F2||C^{\max}$

References

JOHNSON, S. M. (1954): Optimal Two- and Three-Stage Production Schedules with Setup Times Included, *Naval Research Logistics Quaterly*, Vol. 1, pp. 61–68.

MODER, J. J. – PHILLIPS, C. R. – DAVIS, E. W. (1983): Project Management with CPM, PERT and Precedence Diagramming, Van Nostrand Reinhold, New York, 1983.

Deterministic and Stochastic Scheduling, Proceedings of the NATO Advanced Study and Research Institute on Theoretical Approaches to Scheduling Problems held in Durham, England, (1981) ed. by M. A. H. Dempster, J. K. Lenstra, A. H. G. Rinnooy Kan.

PILCHER, R. (1976): Principles of Construction Management, McGraw-Hill Berkshire, England, 1976.

SZWARC, W. (1978): Dominance Conditions for Three-Machine Flow-Shop Problem, *Operations Research* Vol. 26, pp. 203–206.