Journal of the Operations Research
Society of Japan
Vol. 26, No. 3, September 1983

OPTIMAL SCHEDULE IN A GT-TYPE FLOW-SHOP UNDER SERIES-PARALLEL PRECEDENCE CONSTRAINTS

Yasuki Sekiguchi
Hokkaido University

(Received September 24, 1982; Revised April 7, 1983)

Abstract The following two-machine flow-shop scheduling problem is solved. Given jobs are classified into groups, and each machine needs some setup before the first job in a group is started processing. If a job in a group is started its process, all jobs in the group must be finished before a job in another group is processed. Each job may have a specified lag time between machines. Moreover, a series-parallel precedence relation may be specified among groups. Find inter- and intra-group schedules minimizing the total elapsed times on both machines. The proposed method for this problem is based on a new definition of composite jobs and Sidney's theory about series-parallel algorithms. The proposed method can also solve a version of the problem where each job has setup times not included in the processing times and a series-parallel precedence relation is specified among jobs in a group.

1. Introduction

Group technology (GT) is one of the techniques for improving efficiency of job-shop type production. GT is based on classifying parts according to similarity of geometric shape and size, and/or of production process. A shop where GT is implemented can be modelled as follows.

"Jobs to be processed are classified into groups and all jobs in a particular group need some common setup at each production step." Of course, each job may require a particular setup inherent to it. But, such setup operations can often be regarded as a part of machining operations which require the setups. We will treat such cases first, and the results will afterward be shown applicable for cases where setup at each production step of each job must be regarded as an operation independent of machining operation.

Most scheduling problems are very difficult to solve and efficient solution procedures have been developed for a few of them. Most problems with

such a convenient property consist of one or two production steps. Our problem here is also one in a two-production-step shop.

In the previous paper [18], a scheduling problem in a two-machine job-shop where GT is implemented was discussed, and an efficient procedure for finding schedules which minimize the total elapsed times, i.e. the maximum completion times, on both machines under arbitrary time lags was proposed. For the problem, no precedence constaint among groups was imposed, though it imposed a GT-type constraint that jobs are divided into groups and all jobs in a group must be finished without interruption once processing of a job in the group is started.

On the contrary to this, we will treat a problem where a series-parallel precedence constraint is imposed among groups of jobs. But, the shop discussed is not a general job-shop but a two-machine flow-shop where a time lag between the first and the second machines may be specified for each job. The objective is to minimize the total elapsed times on both machines. Our solution allows arbitrary values for parameters and variables such as the number of groups, the number of jobs in a group, lag times, setup times and series-parallel precedence constraints. Various two-machine problems solved in Johnson [4], Mitten [11]-Johnson [5], Jackson [3], Kurisu [6], Sidney [19], Monma [13], Maggu, et al. [10] and Sekiguchi [18] become special cases of our problem by fixing the parameters and/or variables at some specific values. Furthermore, some interesting and unsolved problems such as flow-shop or job-shop problems where setup times for jobs are specified independently of processing times can also be deduced.

In the two succeeding sections, the problem is stated in detail, some basic notations are defined and the results in the preceding paper [18] are briefly introduced. An objective here is to show that Johnson's rule is applicable for a fairly general situation.

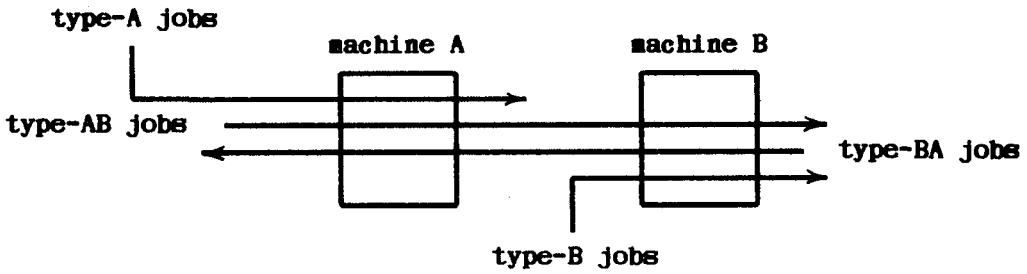
Kurisu [6] first introduced precedence constraints on a flow-shop scheduling problem, and Kurisu [7], Sidney [19], Monma [13] and some others have tried to generalize or to improve his study. Our problem will be analyzed along with Sidney's theory. In that, a concept of composite jobs plays a substantial role. Because our problem is fairly general, composite jobs occasionally have negative processing times, and composition of composite jobs needs more general treatment than that of ordinary jobs. This issue is discussed in 4.

Series-parallel precedence constraints are introduced in 5. A procedure for finding an optimal schedule of our problem imposed series-parallel constraints is proposed. The last section discusses the power of the new solu-

tion and the unsolved issues.

2. Preliminary

Consider a shop with two machines, A and B. The buffer capacity between machines is assumed to be big enough and there is no restriction on the quantity of in-process inventory. A job given to this shop has one of the four technological orders shown in Fig. 2.1 (a). Jobs finished only through machine A (B) are said to be type-A (B), and those finished through machine A



(a) Types of jobs.

group No.	setup time		job No.	processing time		lag time
	A	B		A	B	
i	S_{Ai}	S_{Bi}	i1	A_{i1}	–	–
			i2	A_{i2}	B_{i2}	D_{i2}
			i3	–	B_{i3}	–
			i4	A_{i4}	B_{i4}	D_{i4}
			i5	A_{i5}	–	–

(b) A typical job group: $G_i = \{i1, i2, \cdots, i5\}$.

Fig. 2.1 Illustrations of the problem.

and then machine B are type-AB. Suppose that g job groups are given: $G_i = \{i1, i2, \ldots, in_i\}$, $n_i \geq 1$, $i = 1, 2, \ldots, g$. A_{ij} and B_{ij} are processing times of job ij on machines A and B, respectively. If job ij is type-A (B), only A_{ij} (B_{ij}) is given.

A lag time of a type-AB job is a specified time which must be at least spent from the start of its operation on machine A until its start on machine B (a start lag), and from the completion of its operation on machine A until its completion on machine B (a stop lag). Practically, a start lag of a job may be different from its stop lag, but one of them is actually effective. Without loss of generality, we assume that a lag time D_{ij} may be specified on each type-AB and type-BA job ij. If $D_{ij} = \min\{A_{ij}, B_{ij}\}$, job ij is equivalent to a job without a lag time. Lag times can be applied for expressing overlapping productions, transport times between machines and operations on non-bottleneck machines (see [12] for details), and also for improving lower bounds in a branch and bound method for multistage flow shop problems ([8], [17]).

Denote by S_{Ai} and S_{Bi} setup times of G_i on machines A and B, which must be spent before the first job in the group is started processing. Job groups which includes at least one type-AB job and no type-BA job are called type-AB groups (Fig. 2.1 (b)). Similarly, type-BA groups are defined. A type-A (B) group consists of type-A (B) jobs only.

In the previous paper [18], a procedure which gives an optimal permutation schedule under no precedence constraint when a job group can be any one of those four types. A permutation schedule here is consists of inter-group and intra-group schedules and the orders of type-AB (BA) jobs and type-AB (BA) groups on machine B (A) must be the same as those on machine A (B).

However, all job groups in the present paper are type-AB except in the last section. The problem to be solved is "given g type-AB groups and a precedence constraint among them described as a series-parallel network (see 5.), find an optimal permutation schedule minimizing the maximum completion times on machines A and B."

Let time 0 be the time when machine A can start processing the first job in the given groups and t_0 ($\gtrless 0$) be the time when machine B can start processing the first job. The pair $(0, t_0)$ is called an initial condition [16]. Denote by $T_A(\sigma)$ and $T_B(\sigma)$ the completion times on machines A and B, respec-

tively, when a permutation schedule σ is implemented under an initial condition. $\sigma \cdot ij$ is the schedule made by concatinating job ij after a schedule σ. Calculation of completion times of $\sigma \cdot ij$ depends on the type of ij and is done by (2.1)~(2.3).

(2.1) type-A $T_A(\sigma \cdot ij) = T_A(\sigma) + A_{ij}$, $T_B(\sigma \cdot ij) = T_B(\sigma)$.

(2.2) type-B $T_A(\sigma \cdot ij) = T_A(\sigma)$, $T_B(\sigma \cdot ij) = T_B(\sigma) + B_{ij}$.

type-A $T_A(\sigma \cdot ij) = T_A(\sigma) + A_{ij}$,

(2.3) type-AB

$$T_B(\sigma \cdot ij) = \max \begin{cases} T_A(\sigma) + A_{ij} + D_{ij} \\ T_A(\sigma) + D_{ij} + B_{ij} \\ T_B(\sigma) + B_{ij} \end{cases}$$

When G_i is scheduled just after σ, machines A and B are ready for processing jobs in it at

(2.4) $$T_A(\sigma) + S_{Ai}, \quad T_B(\sigma) + S_{Bi}.$$

Therefore, we assume here for convenience that $T_A(\sigma)$ and $T_B(\sigma)$ in (2.1)~(2.3) are replaced by the respective terms in (2.4) for the first job in G_i.

ϕ denotes an empty schedule. It is possible to understand an initial condition as a pair of completion times of ϕ, i.e.

(2.5) $$T_A(\phi) = 0, \quad T_B(\phi) = t_0$$

The completion times of G_i is the maximum completion times on machines A and B among jobs in G_i. An intra-group schedule which minimize the completion times of G_i scheduled just after σ will minimize also the completion times under the initial condition (2.5), where $t_0 = T_B(\sigma) + S_{Bi} - \{T_A(\sigma) + S_{Ai}\}$. The problem finding such a schedule is called an intra-group scheduling problem. This problem has been solved as follows (cf. theorem 2 in [18]).

Theorem 1. (an optimal intra-group schedule). Let N_A (N_B) be a set of type-A (B) jobs, $N_{AB,\,I}$ ($N_{AB,\,II}$) be a set of type-AB jobs such that $A_{ij} \leqq B_{ij}$ ($A_{ij} \geqq B_{ij}$), and $G_i = N_A \cup N_{AB,\,I} \cup N_{AB,\,II} \cup N_B$, where $N_{AB,\,I} \cap N_{AB,\,II} = \phi$ (i.e. if $A_{ij} = B_{ij}$, job ij must be in exactly one of $N_{AB,\,I}$ and $N_{AB,\,II}$). Define schedules as follows.

σ_A (σ_B): an arbitrary schedule of jobs in N_A (N_B).

$\sigma_{AB,\,I}$: a schedule of jobs in $N_{AB,\,I}$ arranged in nondecreasing order of

D_{ij}.

$\sigma_{AB,\,II}$: a schedule of jobs in $N_{AB,\,II}$ arranged in nonincreasing order of D_{ij}.

An optimal schedule for an intra-group scheduling problem is a permutation schedule σ^* where the schedule on machine A is $\sigma_{AB,\,I}\cdot\sigma_{AB,\,II}\cdot\sigma_A$ and one on machine B is $\sigma_B\cdot\sigma_{AB,\,I}\cdot\sigma_{AB,\,II}$.

Theorem 1 is a composition of the results by Jackson [3] and Mitten [11]. It is important that the intra-group scheduling problem has been solved under arbitrary initial conditions. Thus, the next theorem is almost evident.

Theorem 2. Given an arbitrary inter-group schedule, let σ^* be this schedule with its intra-group schedules determined by theorem 1. Let σ be a schedule with the same inter-group schedule but with intra-group schedules different from those of σ^*. Then, the following hold.

$$T_A(\sigma^*)= T_A(\sigma), \qquad T_B(\sigma^*)\leqq T_B(\sigma).$$

This theorem asserts that, in order to obtain an optimal permutation schedule, intra-group schedules can be fixed at those determined by theorem 1 and finding an optimal inter-group schedule is the remaining problem. This inter-group scheduling problem will be reduced to a problem equivalent to an intra-group scheduling problem by introducing a concept of composite jobs.

Define a triplet $(\alpha_i, \Delta_i, \beta_i)$ corresponding to a schedule of type-AB jobs in G_i. Let T_B be the completion time on machine B of the schedule under initial condition (0, 0). Then,

$$\alpha_i = T_B - \Sigma_{N_B \cup N_{AB}} B_{ij} + S_{Ai} - S_{Bi},$$

$$\beta_i = T_B - \Sigma_{N_A \cup N_{AB}} A_{ij},$$

$$\Delta_i = T_B + S_{Ai} - \max\{\alpha_i, 0\} - \max\{\beta_i, 0\}.$$

This triplet is a set of durations α_i, β_i when exactly one of the machines is busy and a duration Δ_i when both machines are busy (or idle). A typical situation is illustrated in Fig. 2.2. As it is easily understood, β_i can be negative if $\Sigma_{N_A} A_{ij}$ is large and the sign of α_i depends on the values of S_{Bi} and $\Sigma_{N_B} B_{ij}$.

Suppose that G_i in Fig. 2.2 is processed after $T_A(\sigma)$ and $T_B(\sigma)$. The

setup S_{Ai} is surely started at $T_A(\sigma)$, and all operations on machine A can be performed consecutively without intermediate idle time even if $T_B(\sigma)$ is much larger than $T_A(\sigma)$. The setup S_{Bi} is surely started at $T_B(\sigma)$ and operations in N_B can be performed consecutively without intermediate idle time even if $T_A(\sigma)$ is larger than $T_B(\sigma)$. But, notice that operations in N_{AB} may not be started on machine B when all operations in N_B have just been completed, if $T_A(\sigma)$ is not smaller enough than $T_B(\sigma)$. Based on this observation, a composite job is defined as follows.

A composite job J_i corresponding to a schedule of type-AB jobs in G_i is a job with a pair of processing times (α_i, β_i) and without a lag time, whose completion times are determined by the rules shown in Fig. 2.3. From the figure, the readers will understand that (1) a positive α_i and a negative β_i represent processing times of length α_i and $-\beta_i$ on machine A, and a negative α_i and a positive β_i represent those of length $-\alpha_i$ and β_i on machine B, (2) an operation corresponding to α_i (positive or negative) or to negative β_i can be started only if a required machine is available, (3) an operation corresponding to β_i can not be started before the operation corresponding to related α_i is completed, and (4) the second rule in the figure is consistent with (2.3) where $D_{ij} = \min\{A_{ij}, B_{ij}\}$.

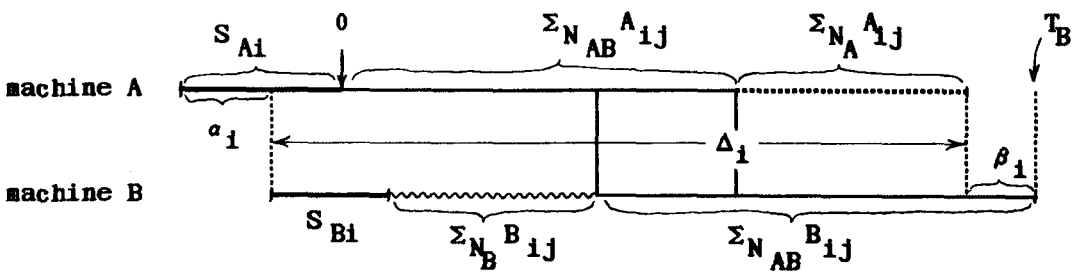


Fig. 2.2 Determination of a composite job.

The definition of composite jobs above is different from that in Kurisu [6] or Sidney [19], where lag times are used as a basic tool. Our definition makes it easy to expand the concept of composite jobs. That is, it is convenient to define composite jobs by lag times if α_i and β_i are positive, because the difference $\Sigma\Delta$ in theorem 3 does not occur. But, this definition does not seem to be able to deal with negative α and β. In order to systematize the theory, it will be benificial to reduce a scheduling problem with lag times ([11] and [5]) into the Johnson problem [4] by introducing composite jobs defined in the present paper (see lines 6~12 of p.786 in [19]).

Denote by σ_J an arbitrary schedule of composite jobs J_i (i= 1, 2, ..., g) corresponding to specific schedules of type-AB jobs in G_i's. Let σ be a

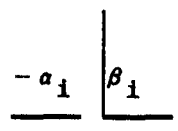
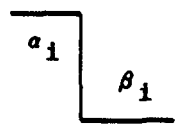
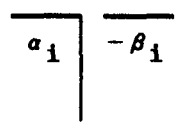
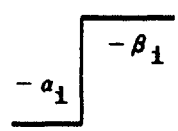
sign	shape	rule for completion time calculation
$\alpha_i \leqq 0$ $\beta_i \geqq 0$		$T_A(\sigma \cdot J_i) = T_A(\sigma)$ $T_B(\sigma \cdot J_i) = \max\left\{\begin{matrix} T_A(\sigma) \\ T_B(\sigma) - \alpha_i \end{matrix}\right\} + \beta_i$
$\alpha_i \geqq 0$ $\beta_i \geqq 0$		$T_A(\sigma \cdot J_i) = T_A(\sigma) + \alpha_i$ $T_B(\sigma \cdot J_i) = \max\left\{\begin{matrix} T_A(\sigma) + \alpha_i \\ T_B(\sigma) \end{matrix}\right\} + \beta_i$
$\alpha_i \geqq 0$ $\beta_i \leqq 0$		$T_A(\sigma \cdot J_i) = T_A(\sigma) + \alpha_i - \beta_i$ $T_B(\sigma \cdot J_i) = \max\left\{\begin{matrix} T_A(\sigma) + \alpha_i \\ T_B(\sigma) \end{matrix}\right\}$
$\alpha_i \leqq 0$ $\beta_i \leqq 0$		$T_A(\sigma \cdot J_i) = T_A(\sigma) - \beta_i$ $T_B(\sigma \cdot J_i) = \max\left\{\begin{matrix} T_A(\sigma) \\ T_B(\sigma) - \alpha_i \end{matrix}\right\}$

Fig. 2.3 Composite jobs and rules for completion time calculation.

permutation schedule with the inter-group schedule σ_J and intra-group schedules used in defining J_i's. The completion times of σ_J and σ are calculated by Fig. 2.3. and (2.1)~(2.3), respectively. Then, the next theorem can be proved [18].

Theorem 3. The following relations hold under arbitrary initial conditions.

$$T_A(\sigma) = T_A(\sigma_J) + \Sigma_{i=1}^{g} \Delta_i.$$
$$T_B(\sigma) = T_B(\sigma_J) + \Sigma_{i=1}^{g} \Delta_i.$$

This theorem asserts that the problem of finding an optimal inter-group schedule under intra-group schedules specified for job groups is equivalent to that of finding an optimal composite job schedule, where the composite job for a group corresponds to the specified intra-group schedule. An ordinary type-AB job is equivalent to the second type in Fig. 2.3 and its completion time is determined by the same formula as (2.3), where $D_{ij} = \min\{A_{ij}, B_{ij}\}$. It is understood that the problem of finding an optimal composite job schedule is a generalization of the classic flow-shop problem (or the intra-group scheduling problem) in Bellman [1] and Johnson [4]. Notice that the computational rules in Fig. 2.3 are equivalent to

(2.6)
$$T_A(\sigma \cdot J_i) = T_A(\sigma) + \max\{\alpha_i, 0\} - \min\{0, \beta_i\},$$
$$T_B(\sigma \cdot J_i) = \max\left\{\begin{matrix} T_A(\sigma) + \max\{\alpha_i, 0\} \\ T_B(\sigma) - \min\{0, \alpha_i\} \end{matrix}\right\} + \max\{\beta_i, 0\}.$$

3. Dominance Relations among Composite Jobs

The author resolved the composite job scheduling problem in the previous paper (theorem 8 in [18]). The result and its proof will be given in a simpler form.

Definition (dominance relation). When one of the following relations holds. J_i is said to dominate J_j, and denoted by $J_i \leqq J_j$.

(3.1) $\alpha_i \leqq \beta_i$, $\alpha_j \leqq \beta_j$, $\alpha_i \leqq \alpha_j$,

(3.2) $\alpha_i \leqq \beta_i$, $\alpha_j \geqq \beta_j$,

(3.3) $\alpha_i \geqq \beta_i$, $\alpha_j \geqq \beta_j$, $\beta_i \geqq \beta_j$.

Both $\alpha_i \leqq \beta_i$ and $\alpha_i \geqq \beta_i$ are true if $\alpha_i = \beta_i$ holds, but we assume that $\alpha_i = \beta_i$ is never interpreted as $\alpha_i \leqq \beta_i$ in one place and as $\alpha_i \geqq \beta_i$ in another place. That is, if $\alpha_i = \beta_i$ is interpreted as $\alpha_i \leqq \beta_i$ when composite job i is first compared to another one, it must be interpreted as $\alpha_i \leqq \beta_i$ in all comparisons. This assumption is called a unique interpretation assumption (UIA). Notice that UIA does not prohibit from interpreting $\alpha_i = \beta_i$ as $\alpha_i \geqq \beta_i$ when $\alpha_j = \beta_j$ $(i \neq j)$ is interpreted as $\alpha_j \leqq \beta_j$.

Theorem 4. The dominance relation $\leqq$ is transitive under UIA.

Proof: Suppose $J_i \leqq J_j$ and $J_j \leqq J_k$. When J_i and J_j satisfy (3.1), J_j and J_k must satisfy (3.1) or (3.2). In the former case, J_i and J_k satisfies (3.1), and in the latter case (3.2). When J_i and J_j satisfy (3.2), J_j and J_k must satisfy (3.3). Then, J_i and J_k satisfy (3.2). When J_i and J_j satisfy (3.3), J_j and J_k must also satisfy (3.3). Thus, J_i and J_k satisfy (3.3).

Theorem 5. If $J_i \leqq J_j$, the following are true under arbitrary initial conditions.

$$T_A(J_i \cdot J_j) = T_A(J_j \cdot J_i),\quad T_B(J_i \cdot J_j) \leqq T_B(J_j \cdot J_i).$$

Proof:

$$\begin{aligned} T_A(J_i \cdot J_j) &= T_A(J_i) + \max\{\alpha_j, 0\} - \min\{0, \beta_j\} \\ &= \max\{\alpha_i, 0\} - \min\{0, \beta_i\} + \max\{\alpha_j, 0\} - \min\{0, \beta_j\} \\ &= T_A(J_j, J_i). \end{aligned}$$

$$T_B(J_i \cdot J_j) = \max\left\{\begin{matrix} T_A(J_i) + \max\{\alpha_j, 0\} \\ T_B(J_i) - \min\{0, \alpha_j\} \end{matrix}\right\} + \max\{\beta_j, 0\}$$

$$= \max\left\{\begin{matrix} \max\{\alpha_i, 0\} - \min\{0, \beta_i\} + \max\{\alpha_j, 0\} \\ \max\left\{\begin{matrix} \max\{\alpha_i, 0\} \\ T_B(\phi) - \min\{0, \alpha_i\} \end{matrix}\right\} + \max\{\beta_i, 0\} - \min\{0, \alpha_j\} \end{matrix}\right\}$$

$$+ \max\{\beta_j, 0\}$$

$$= \max\left\{\begin{matrix} \max\{\alpha_i, 0\} - \min\{0, \beta_i\} + \max\{\alpha_j, 0\} + \max\{\beta_j, 0\} \\ \max\{\alpha_i, 0\} + \max\{\beta_i, 0\} - \min\{0, \alpha_j\} + \max\{\beta_j, 0\} \\ t_0 - \min\{0, \alpha_i\} + \max\{\beta_i, 0\} - \min\{0, \alpha_j\} + \max\{\beta_j, 0\} \end{matrix}\right\}$$

Similarly,

$$T_B(J_j \cdot J_i) = \max \begin{cases} \max\{\alpha_j, 0\} - \min\{0, \beta_j\} + \max\{\alpha_i, 0\} + \max\{\beta_i, 0\} \\ \max\{\alpha_j, 0\} + \max\{\beta_j, 0\} - \min\{0, \alpha_i\} + \max\{\beta_i, 0\} \\ t_0 - \min\{0, \alpha_j\} + \max\{\beta_j, 0\} - \min\{0, \alpha_i\} + \max\{\beta_i, 0\} \end{cases}$$

Let ij1, ij2, ij3 be the first, second and third terms in $T_B(J_i \cdot J_j)$, respectively. Define similarly ji1, ji2 and ji3. Evidently, ij3=ji3. If (3.1) holds,

$$ij1 - ji2 = \max\{\alpha_i, 0\} + \min\{0, \alpha_i\} - \min\{0, \beta_i\} - \max\{\beta_i, 0\}$$
$$= \alpha_i - \beta_i \leqq 0$$
$$ij2 - ji2 = \max\{\alpha_j, 0\} + \min\{0, \alpha_j\} - \min\{0, \alpha_j\} - \max\{\alpha_j, 0\}$$
$$= \alpha_j - \alpha_j \leqq 0.$$

When (3.2) holds, then

$$ij1 - ji2 = \alpha_i - \beta_i \leqq 0,$$
$$ij2 - ji1 = -\min\{0, \alpha_j\} - \max\{\alpha_j, 0\} + \max\{\beta_j, 0\} + \min\{0, \beta_j\}$$
$$= -\alpha_j + \beta_j \leqq 0.$$

When (3.3) holds, the following relations are true.

$$ij1 - ji1 = -\min\{0, \beta_i\} - \max\{\beta_i, 0\} + \max\{\beta_j, 0\} + \min\{0, \beta_j\}$$
$$= -\beta_i + \beta_j \leqq 0,$$
$$ij2 - ji1 = -\alpha_j + \beta_j \leqq 0.$$

Thus, $T_B(J_i \cdot J_j) \leqq T_B(J_j \cdot J_i)$ is true.

Theorem 5 gives the following theorem together with theorem 4.

Theorem 6 (a generalized Johnson's rule). A composite job schedule is optimal if any part of it does not conflict with the dominance relation under UIA.

As a result, if no precedence constraint is imposed, optimal inter- and intra-group schedules can be obtained by the following procedure. First, find an optimal intra-group schedule for each job group by theorem 1. Second, define composite jobs corresponding to the optimal intra-group schedules of job groups. Finally, find an optimal inter-group schedule as an optimal composite job schedule given by theorem 6.

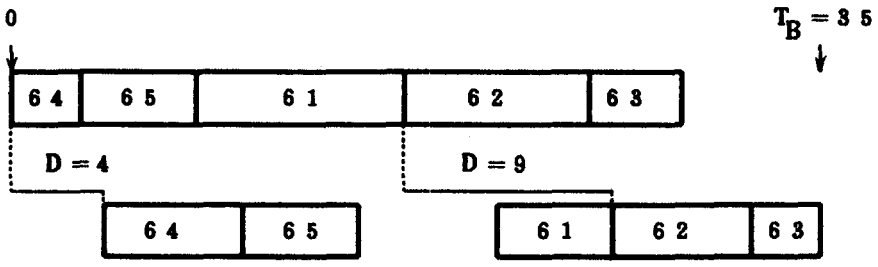
Example 1. (No precedence constraint is imposed.)

Suppose that job groups and jobs in Table 3.1 are given. Composite jobs

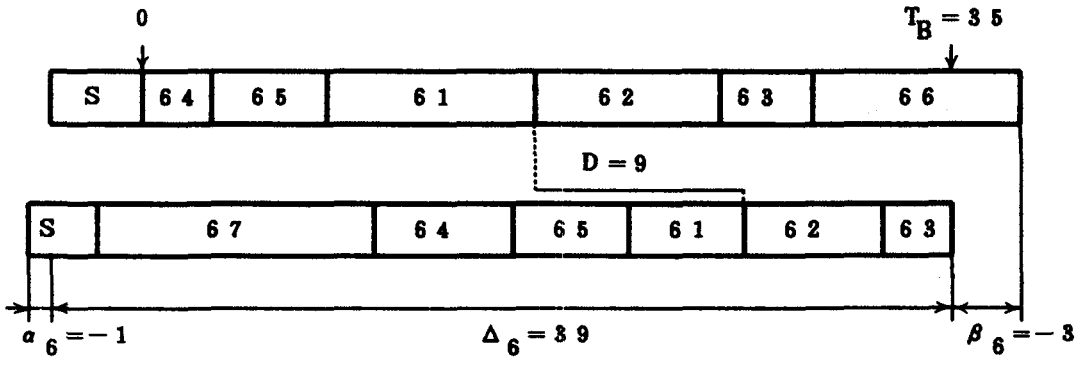
are determined as shown in Table 3.2. For example, Fig.3.1 is the Gantt's chart used in determining the composite job corresponding to G_6. Applying theorem 6 to the composite jobs in Table 3.2, we will get the following composite job schedule minimizing the completion times: $J_4 \cdot J_7 \cdot J_5 \cdot J_2 \cdot J_1 \cdot J_3 \cdot J_6$. Therefore, by substituting intra-group schedules in Table 3.2, an optimal permutation schedule is given as

machine A: $S_{A4} \cdot 41 \cdot S_{A7} \cdot 72 \cdot 71 \cdot 74 \cdot S_{A5} \cdot 52 \cdot 53 \cdot 51 \cdot S_{A2} \cdot 21 \cdot 22 \cdot S_{A1} \cdot 11 \cdot 12 \cdot 13$
$\cdot S_{A3} \cdot 32 \cdot 31 \cdot 33 \cdot 34 \cdot 35 \cdot S_{A6} \cdot 64 \cdot 65 \cdot 61 \cdot 62 \cdot 63 \cdot 66,$

machine B: $S_{B4} \cdot 42 \cdot 41 \cdot S_{B7} \cdot 73 \cdot 72 \cdot 71 \cdot S_{B5} \cdot 54 \cdot 52 \cdot 53 \cdot 51 \cdot S_{B2} \cdot 21 \cdot 22 \cdot S_{B1} \cdot 11 \cdot 12 \cdot 13$
$\cdot S_{B3} \cdot 32 \cdot 31 \cdot 33 \cdot 34 \cdot S_{B6} \cdot 67 \cdot 64 \cdot 65 \cdot 61 \cdot 62 \cdot 63.$



(a) Calculation of T_B.



(b) Determination of α, β and Δ.

Fig. 3.1 Determination of the composite job to G_6.

Table 3.1 An example : List of jobs.

group No.	setup time		job No.	processing time		lag time
	A	B		A	B	
1	5	6	1 1 1 2 1 3	8 9 9	5 1 9	8 6 3
2	7	3	2 1 2 2	6 5	2 3	9 1
3	1	5	3 1 3 2 3 3 3 4 3 5	4 6 4 6 4	6 6 1 3 –	5 2 5 1 –
4	9	6	4 1 4 2	4 –	8 1 3	6 –
5	9	8	5 1 5 2 5 3 5 4	2 2 4 –	9 6 6 5	6 2 2 –
6	4	3	6 1 6 2 6 3 6 4 6 5 6 6 6 7	9 8 4 3 5 9 –	5 6 3 6 5 – 1 2	9 4 1 4 6 – –
7	9	5	7 1 7 2 7 3 7 4	4 6 – 9	6 6 1 3 –	5 4 – –

Table 3.2 An example : Composite jobs.

group No.	intra-group schedules	α	β	Δ
1	A : 1 1 · 1 2 · 1 3 B : 1 1 · 1 2 · 1 3	1 6	6	1 5
2	A : 2 1 · 2 2 B : 2 1 · 2 2	1 7	7	1
3	A : 3 2 · 3 1 · 3 3 · 3 4 · 3 5 B : 3 2 · 3 1 · 3 3 · 3 4	2	− 2	2 1
4	A : 4 1 B : 4 2 · 4 1	− 4	1 0	1 3
5	A : 5 2 · 5 3 · 5 1 B : 5 4 · 5 2 · 5 3 · 5 1	− 2	1 5	1 7
6	A : 6 4 · 6 5 · 6 1 · 6 2 · 6 3 · 6 6 B : 6 7 · 6 4 · 6 5 · 6 1 · 6 2 · 6 3	− 1	− 3	3 9
7	A : 7 2 · 7 1 · 7 4 B : 7 3 · 7 2 · 7 1	− 4	− 2	2 6

4. Composition of Composite Jobs and Dominance Relation

In the next section, we discuss a procedure determining an optimal composite job schedule under some precedence relation. In doing this, compositions of composite jobs play the main role, and so we define a composition of composite jobs and show that this composition conserves the dominance relation between original composite jobs.

G_1 and G_2 are arbitrary type-AB groups. Suppose that triplets $(\alpha_1, \Delta_1, \beta_1)$ and $(\alpha_2, \Delta_2, \beta_2)$ are determined for them under some given intra-group schedules and suppose also that composite jobs J_1: (α_1, β_1) and J_2: (α_2, β_2) are defined. A suppositional job group G_J is defined as follows.

$G_J = \{J_1', J_2\}$, $S_{AJ} = 0$, $S_{BJ} = -\min\{0, \alpha_1\}$, J_1': $(\max\{\alpha_1, 0\}, \beta_1)$

In other words, J_1' is J_1 neglected its negative α_1 (if any), and the neglected α_1 is treated as the setup time on machine B of G_J. The completion time of the suppositional composite job J_1' is assumed to be calculated according to the rules in Fig. 2.3.

Let T_B be the completion time of schedule $J_1' \cdot J_2$ under initial condition (0, 0). Then, a triplet $(\alpha_{12}, \Delta_{12}, \beta_{12})$ corresponding to $J_1' \cdot J_2$ is defined as follows.

$$\alpha_{12} = T_B - [\max\{\beta_1, 0\} - \min\{0, \alpha_2\} + \max\{\beta_2, 0\}] - S_{BJ},$$

$$\beta_{12} = T_B - [\max\{\alpha_1, 0\} - \min\{0, \beta_1\} + \max\{\alpha_2, 0\} - \min\{0, \beta_2\}],$$

$$\Delta_{12} = \Delta_1 + \Delta_2 + T_B - \max\{\alpha_{12}, 0\} - \max\{\beta_{12}, 0\}.$$

The terms in $[\cdot]$ of α_{12} and β_{12} are the total length of processing times of J_1' and J_2 on machines B and A, respectively. Thus, this definition of a triplet is equivalent to that in the preceding section. Therefore, a composite job J_{12} corresponding to a schedule $J_1 \cdot J_2$ of the original composite jobs is defined as a pair $(\alpha_{12}, \beta_{12})$ whose completion times are calculated by Fig. 2.3.

By calculation similar to that in the proof of theorem 5, we get

$$T_A(J_1' \cdot J_2) = \max\{\alpha_1, 0\} - \min\{0, \beta_1\} + \max\{\alpha_2, 0\} - \min\{0, \beta_2\}$$

$$T_B(J_1' \cdot J_2) = T_B \text{ (by definition)}$$

$$= \max\left\{\begin{array}{l} \max\{\alpha_1, 0\} - \min\{0, \beta_1\} + \max\{\alpha_2, 0\} + \max\{\beta_2, 0\} \\ \max\{\alpha_1, 0\} + \max\{\beta_1, 0\} - \min\{0, \alpha_2\} + \max\{\beta_2, 0\} \end{array}\right\}.$$

Substitute this into α_{12} and β_{12}, then finally we get

$$\alpha_{12} = \alpha_1 + \max\{\alpha_2 - \beta_1, 0\},$$
$$\beta_{12} = \beta_2 + \max\{\beta_1 - \alpha_2, 0\}. \tag{4.1}$$

Theorem 7. Under an arbitrary initial condition, the following are true.

$$T_A(J_{12}) + \Delta_{12} = T_A(J_1 \cdot J_2) + \Delta_1 + \Delta_2,$$
$$T_B(J_{12}) + \Delta_{12} = T_B(J_1 \cdot J_2) + \Delta_1 + \Delta_2.$$

Proof:

$$\begin{aligned} T_A(J_{12}) + \Delta_{12} &= \max\{\alpha_{12}, 0\} - \min\{0, \beta_{12}\} + \Delta_{12} && \text{(by Fig. 2.3)} \\ &= \Delta_1 + \Delta_2 + T_B - \min\{0, \beta_{12}\} - \max\{\beta_{12}, 0\} && \text{(by definition)} \\ &= \Delta_1 + \Delta_2 + T_B - \beta_{12} \\ &= \Delta_1 + \Delta_2 + [\max\{\alpha_1, 0\} - \min\{0, \beta_1\} \\ &\qquad + \max\{\alpha_2, 0\} - \min\{0, \beta_2\}] && \text{(by definition)} \end{aligned}$$

$$T_B(J_{12}) + \Delta_{12} = \max\left\{\begin{matrix}\max\{\alpha_{12}, 0\} \\ t_0 - \min\{0, \alpha_{12}\}\end{matrix}\right\} + \max\{\beta_{12}, 0\} + \Delta_{12}$$

$$= \max\left\{\begin{matrix} T_B \\ t_0 - \min\{0, \alpha_{12}\} - \max\{\alpha_{12}, 0\} + T_B\end{matrix}\right\} + \Delta_1 + \Delta_2$$

(by definition)

$$= \max\left\{\begin{matrix} T_B \\ t_0 - \alpha_{12} + T_B\end{matrix}\right\} + \Delta_1 + \Delta_2$$

$$= \max\left\{\begin{matrix}\max\{\alpha_1, 0\} - \min\{0, \beta_1\} + \max\{\alpha_2, 0\} + \max\{\beta_2, 0\} \\ \max\{\alpha_1, 0\} + \max\{\beta_1, 0\} - \min\{0, \alpha_2\} + \max\{\beta_2, 0\} \\ t_0 + \max\{\beta_1, 0\} - \min\{0, \alpha_1\} + \max\{\beta_2, 0\} - \min\{0, \alpha_2\}\end{matrix}\right\}$$
$$+ \Delta_1 + \Delta_2 \qquad \text{(by definition.)}$$

Calculate similarly $T_A(J_1 \cdot J_2) + \Delta_1 + \Delta_2$ and $T_B(J_1 \cdot J_2) + \Delta_1 + \Delta_2$, then the proof is completed.

Remember that theorem 3 guarantees the following. "Given g job groups with a fixed intra-group schedule for each group, an optimal inter-group schedule can be obtained by solving a composite job scheduling problem induced by exchanging each group with the respective composite job." A similar property of composite jobs can be proved by using theorem 7 repeatedly: "Given some composite jobs with a fixed schedule of a subset of the composite jobs, an optimal composite job schedule can be obtained by solving a reduced composite job scheduling problem induced by exchanging the subset with the re-

spective composite job."

As discussed in the preceding section, an optimal schedule can be obtained by arranging composite jobs according to the dominance relation, if no precedence relation is imposed. Now, suppose that four composite jobs, J_1, J_2, J_3 and J_4 are given, and $J_3 \leqq J_1 \leqq J_2 \leqq J_4$ holds. Then an optimal schedule is $J_3 \cdot J_1 \cdot J_2 \cdot J_4$. Suppose that we need an optimal schedule which includes schedule $J_1 \cdot J_2$. $J_3 \cdot J_1 \cdot J_2 \cdot J_4$ remains optimal. Thus, theorem 7 suggests that $J_3 \cdot J_{12} \cdot J_4$ is an optimal schedule when $J_1 \cdot J_2$ is replaced by J_{12}. It is natural to expect $J_3 \leqq J_{12} \leqq J_4$. Notice that this is not evident from theorem 6 because it merely gives a sufficient condition of optimal solutions. The next theorem shows this property in a stronger form.

Theorem 8. Let J_{12} be the composite job corresponding to $J_1 \cdot J_2$.

If $J_1 \leqq J_2$, then $J_1 \leqq J_{12} \leqq J_2$.

If $J_1 \geqq J_2$, then $J_1 \geqq J_{12} \geqq J_2$.

Proof: The first and the second parts of the theorem can be proved by the similar procedures. The second part is proved here.

$J_2 \leqq J_1$ implies that one of (3.1)~(3.3) with $i=2$ and $j=1$ must be true. If (3.1) is true, $\alpha_2 \leqq \alpha_1 \leqq \beta_1$. By (4.1), $\alpha_{12}=\alpha_1$ and $\beta_{12}=\beta_1+(\beta_2-\alpha_2) \geqq \beta_1$. This implies $\alpha_{12} \leqq \beta_{12}$. Then, $\alpha_1=\alpha_{12}$ implies $J_{12} \leqq J_1$ and $\alpha_2 \leqq \alpha_{12}=\alpha_1$ implies $J_2 \leqq J_{12}$, because (3.1) holds. If (3.3) is true, $\alpha_{12}=\alpha_1+\alpha_2-\beta_1 \geqq \alpha_2$ and $\beta_{12}=\beta_2$. Moreover, $\alpha_{12} \geqq \beta_{12}$ because $\beta_2 \leqq \alpha_2$. By (3.3), both $J_2 \leqq J_{12}$ and $J_{12} \leqq J_1$ are true. Now assume (3.2). If $\beta_1 \leqq \alpha_2$, $\alpha_{12}=\alpha_1+\alpha_2-\beta_1 \geqq \alpha_2$ and $\beta_{12}=\beta_2$. Thus, $J_2 \leqq J_{12}$ by (3.1) and $J_{12} \leqq J_1$ by (3.2) when $\alpha_{12} \leqq \beta_{12}$, and $J_2 \leqq J_{12}$ by (3.2) and $J_{12} \leqq J_2$ by (3.3) when $\alpha_{12} \geqq \beta_{12}$ (notice that $\beta_1 \leqq \alpha_2 \leqq \beta_2=\beta_{12}$). If $\beta_1 \geqq \alpha_2$, $\alpha_2 \leqq \beta_1 \leqq \alpha_1$. Therefore, $\alpha_{12}=\alpha_1$ and $\beta_{12}=\beta_2+\beta_1-\alpha_2 \geqq \beta_1$. Thus, if $\alpha_{12} \leqq \beta_{12}$, $J_2 \leqq J_{12}$ by (3.1) and $J_{12} \leqq J_1$ by (3.2). If $\alpha_{12} \geqq \beta_{12}$, $J_2 \leqq J_{12}$ by (3.2) and $J_{12} \leqq J_1$ by (3.3). The proof has been completed.

5. Optimal Schedule under Series-Parallel Precedence Relation

Consider a network R whose nodes correspond to job groups and arcs to precedence relations among them. G_i is said to precede G_j if there is a directed path from G_i to G_j. If there is an arc starting at G_i and ending at G_j, G_i is said to precede G_j directly. Theorem 3 guarantees that an inter-group schedule minimizing the maximum completion times under R is the same as the minimum total elapsed time schedule of a problem induced by replacing each job group in R by a composite job corresponding to its given intra-group schedule. Therefore, J_i is used for G_i below.

A chain in R is a directed path $J_i \rightarrow J_j \rightarrow \ldots \rightarrow J_k$ such that for each composite job J_q not included in the path, one of the following relations holds: (1) J_q precedes all jobs in the path, (2) J_q succeeds to all jobs in the path, (3) J_q has no precedence relation with anyone in the path. Simply, a chain is a directed path with no intermediate branch. Two networks are said to be parallel if they are disconnected from each other. Theorem 9 and 10 have been shown for ordinary type-AB jobs by Sidney [19]. We will extend them for composite jobs.

Theorem 9. Suppose that $J_i \geqq J_j$ holds under UIA for an arc $J_i \rightarrow J_j$ which is a chain. Then, there is a schedule in which J_i is processed just before J_j, and which minimizes the maximum completion times under R.

Proof: Let an arbitrary optimal schedule be $\sigma_{J1} \cdot J_i \cdot \sigma_{J2} \cdot J_j \cdot \sigma_{J3}$. Let J_σ and Δ_σ be the composite job and the constant corresponding to σ_{J2}. Composite jobs in σ_{J2} have no precedence relation with J_i and J_j because of the definition of a chain. By theorem 7, two relations below hold.

$$T_A(\sigma_{J1} \cdot J_i \cdot \sigma_{J2} \cdot J_j) = T_A(\sigma_{J1} \cdot J_i \cdot J_\sigma \cdot J_j) + \Delta_\sigma,$$

$$T_B(\sigma_{J1} \cdot J_i \cdot \sigma_{J2} \cdot J_j) = T_B(\sigma_{J1} \cdot J_i \cdot J_\sigma \cdot J_j) + \Delta_\sigma$$

If $J_\sigma \leqq J_j$, then $J_\sigma \leqq J_i$ by transitivity of $\leqq$ (theorem 4). By theorem 5,

$$T_A(\sigma_{J1} \cdot J_\sigma \cdot J_i \cdot J_j) = T_A(\sigma_{J1} \cdot J_i \cdot J_\sigma \cdot J_j),$$

$$T_B(\sigma_{J1} \cdot J_\sigma \cdot J_i \cdot J_j) \leqq T_B(\sigma_{J1} \cdot J_i \cdot J_\sigma \cdot J_j).$$

If $J_j \leqq J_\sigma$, then again by thorem 5,

$$T_A(\sigma_{J1} \cdot J_i \cdot J_j \cdot J_\sigma) = T_A(\sigma_{J1} \cdot J_i \cdot J_\sigma \cdot J_j),$$
$$T_B(\sigma_{J1} \cdot J_i \cdot J_j \cdot J_\sigma) \leq T_B(\sigma_{J1} \cdot J_i \cdot J_\sigma \cdot J_j).$$

Theorem 10. Suppose that $J_i \geqq J_j \geqq \ldots \geqq J_k$ holds under UIA for a chain $J_i \rightarrow J_j \rightarrow \ldots \rightarrow J_k$ in a network R. Then, there is a schedule including $J_i \cdot J_j \cdot \ldots \cdot J_k$ as a consecutive subschedule, which minimizes the maximum completion times under R.

Proof: Easily completed by a repeated use of theorems 9 and 8.

Suppose that a chain in R satisfies the condition of theorem 10. Make a composite job corresponding to a (sub) schedule determined by the chain. Replace the chain in R by a node related to the composite job, and also replace composite jobs in the chain by the composite job. Then, a reduced problem with fewer composite jobs and a simpler precedence relation is obtained. Theorems 7 and 10 assert that the original problem is solved if the reduced problem is solved. Reduction of a problem can be repeated as long as there is a chain satisfying the condition of theorem 10 (or 9).

Consider that we finally obtain a parallel chain network (some chains parallel to each other) by repeating the reduction, where there is no conflicts between precedence and dominance relations. (For this to happen, the original network must already be a parallel chain network.) In this case, a schedule given by the generalized Johnson's rule naturally satisfies the parallel chain precedence relation. Thus, it is easy to understand that the following procedure gives a schedule minimizing the maximum completion times under a parallel chain precedence relation.

Parallel Chain Algorithm

(1) Find an arc in a given parallel chain network, whose direction opposes the dominance relation $\leq$ under UIA, and replace the arc and the job groups at its both ends with a composite job corresponding to a schedule determined by the arc. Repeat this operation until no arc conflicts with the dominance relation in the reduced network.

(2) Find an optimal composite job schedule on the reduced network by the generalized Johnson's rule.

When the second step of the parallel chain algorithm is completed, we have a schedule of composite jobs. Substitute the respective schedule of original jobs into each composite jobs in the schedule, and the result is an optimal permutation schedule.

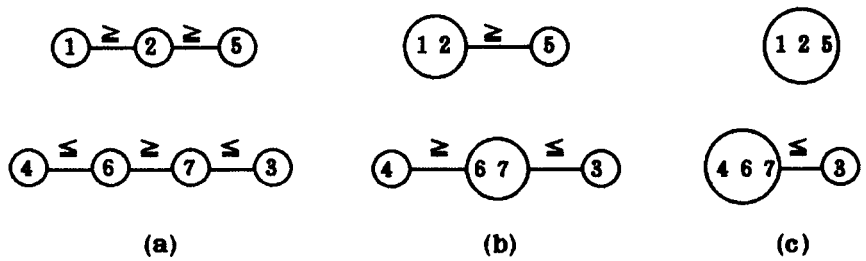
Example 2. (A parallel chain precedence relation is imposed.)

Suppose that the parallel chain precedence relation of Fig. 5.1 (a) is imposed on job groups in example 1 (cf. tables 3.1 and 3.2). In Fig. 5.1 (a), dominance relations among composite jobs are also indicated, and the arcs $J_1 \to J_2$ and $J_6 \to J_7$ can be replaced by composite jobs J_{12} and J_{67}, respectively. The reduced network is Fig. 5.1 (b). J_{12} and J_{67} are shown in Fig. 5.1 (d). Based on the dominance relations shown in Fig. 5.1 (b), the arcs $J_{12} \to J_5$ and $J_4 \to J_{67}$ are replaced by J_{125} and J_{467}, respectively (Fig. 5.1 (c)). There is no arc which conflicts to a dominance relation, and the generalized Johnson's rule is used. The optimal composite job schedule is $J_{467} \cdot J_{125} \cdot J_3$ with the maximum completion times 31 on machine A and 51 on machine B. Add Δ's on them, then the real maximum completion times on machines A and B are 174 and 194 (cf. theorems 7 and 3).

A parallel chain subnetwork in a network is one

(a) that is composed of parallel chains, and

(b) the first (last) node of every chain has the same set of preceding (succeeding) nodes.



J_{12}: $\alpha_{12}=27$, $\beta_{12}=7$, $\Delta_{12}=6+15+1=22$

J_{67}: $\alpha_{67}=-1$, $\beta_{67}=-1$, $\Delta_{67}=39+26+4=69$

J_{125}: $\alpha_{125}=27$, $\beta_{125}=24$, $\Delta_{125}=22+17=39$

J_{467}: $\alpha_{467}=-4$, $\beta_{467}=10$, $\Delta_{467}=13+69+1=83$

(d)

Fig. 5.1 Example 2.

A series-parallel network is one that can be reduced into a chain by replacing repeatedly a parallel chain subnetwork with a chain. Fig. 5.2 illustrates examples. Readers who are interested in a constructive definition of a series-parallel network may refer to Lawler [9]. The chains $J_3 \rightarrow J_6$ and $J_4 \rightarrow J_7$ in the upper network in Fig. 5.2 constitute a parallel chain subnetwork. The chains $J_2 \rightarrow J_5$, $J_3 \rightarrow J_6$ and $J_4 \rightarrow J_7$ are parallel to each other, but they do not constitute a parallel chain subnetwork, since the set of successors for J_5 is not equal to that for J_6 and J_7.

Sidney [19] has shown the next theorem based only on the following three properties:

(I) An optimal schedule is a permutation.

(II) An optimal schedule under arbitrary initial conditions can be obtained for a problem with a parallel chain precedence relation by the parallel chain algorithm.

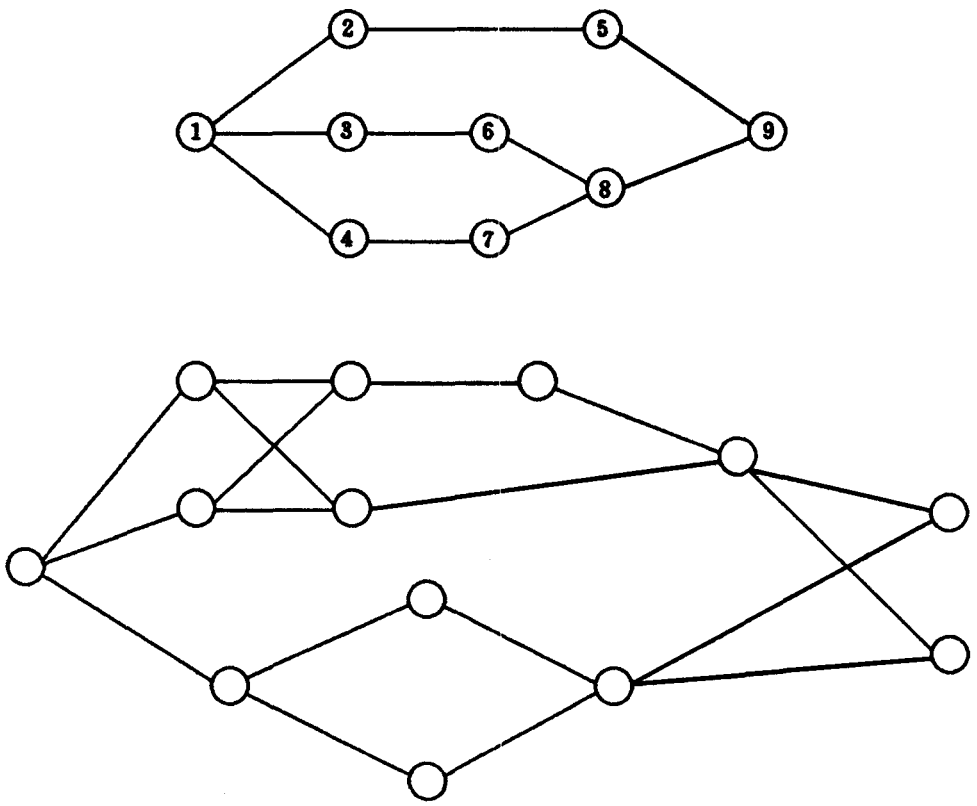


Fig. 5.2 Series-parallel networks.

(III) Let $\sigma_1 \cdot \sigma_2 \cdot \sigma_3$ be an optimal schedule to an original problem. Suppose that σ_2' is an optimal schedule of a problem obtained by restricting the original problem to composite jobs in σ_2. Then, $\sigma_1 \cdot \sigma_2' \cdot \sigma_3$ is also an optimal schedule of the original problem.

Theorem 11. Consider a problem finding an optimal schedule under a precedence relation R. Suppose that R includes a parallel chain subnetwork. Let σ be an optimal schedule of a problem obtained by restricting the original problem to composite jobs in the subnetwork, which is given by the parallel chain algorithm. Then, there is an optimal schedule of the original problem which includes σ as a (not necessarily consecutive) subschedule.

Our problem evidently satisfies (I) and (II). (III) must also be satisfied, since the completion times of σ_3 are nondecreasing functions of its starting times. Thus, theorem 11 can be applied also for our problem. By this theorem, a parallel chain subnetwork in R can be replaced with a chain corresponding to σ.

Series-Parallel Algorithm

(1) Replace a parallel chain subnetwork of R with a chain corresponding to an optimal schedule obtained by the parallel chain algorithm applied for the subnetwork.
(2) Repeat operation (1) as long as parallel chain subnetworks exist.

Theorem 12. The series-parallel algorithm gives a schedule minimizing the maximum completion times on both machines for arbitrary initial conditions under a series-parallel precedence relation.

This theorem can easily be proved by using theorem 11 and the definition of a series-parallel network.

Example 3. (A series-parallel precedence relation is imposed.)

Consider the problem in example 1 and suppose that a precedence relation among job groups is described by the series-parallel network in Fig. 5.3 (a). Calculation by the series-parallel algorithm may proceed as follows.

There are two parallel chain subnetworks in Fig. 5.3 (a), i.e. one is a chain $J_2 \rightarrow J_5$, and the other is a subnetwork composed of J_3 and J_4. For the chain $J_2 \rightarrow J_5$, it is verified that $J_5 \rightarrow J_2$. Thus, according to the first step of the parallel-chain algorithm, this chain is replaced with a composite job J_{25}: $(\alpha_{25}, \beta_{25})$ defined as

$\alpha_{25} = 17 + \max\{-2-7, 0\} = 17$ and $\beta_{25} = 15 + \max\{7+2, 0\} = 24$.

The constant of this composite job is $\Delta_{25}= 18$. Since the chain is reduced to a node, the second step is skipped. Next, the parallel chain algorithm may be applied for the parallel chain subnetwork composed of J_3 and J_4. Then, the first step is skipped, and by the second step this subnetwork is replaced with a chain $J_4 \rightarrow J_3$ corresponding to the schedule $J_4 \cdot J_3$ determined by the generalized Johnson's rule. The reduced network is shown in Fig. 5.3 (b). The remaining parallel chain is composed of J_{25} and a chain $J_4 \rightarrow J_3 \rightarrow J_6$, and all arcs are consistent with dominance relations. Thus $J_4 \cdot J_{25} \cdot J_3 \cdot J_6$ is obtained by the generalized Johnson's rule, and the final reduced network is shown in Fig. 5.3 (c). The optimal inter-group schedule is $J_1 \cdot J_4 \cdot J_2 \cdot J_5 \cdot J_3 \cdot J_6 \cdot J_7$.

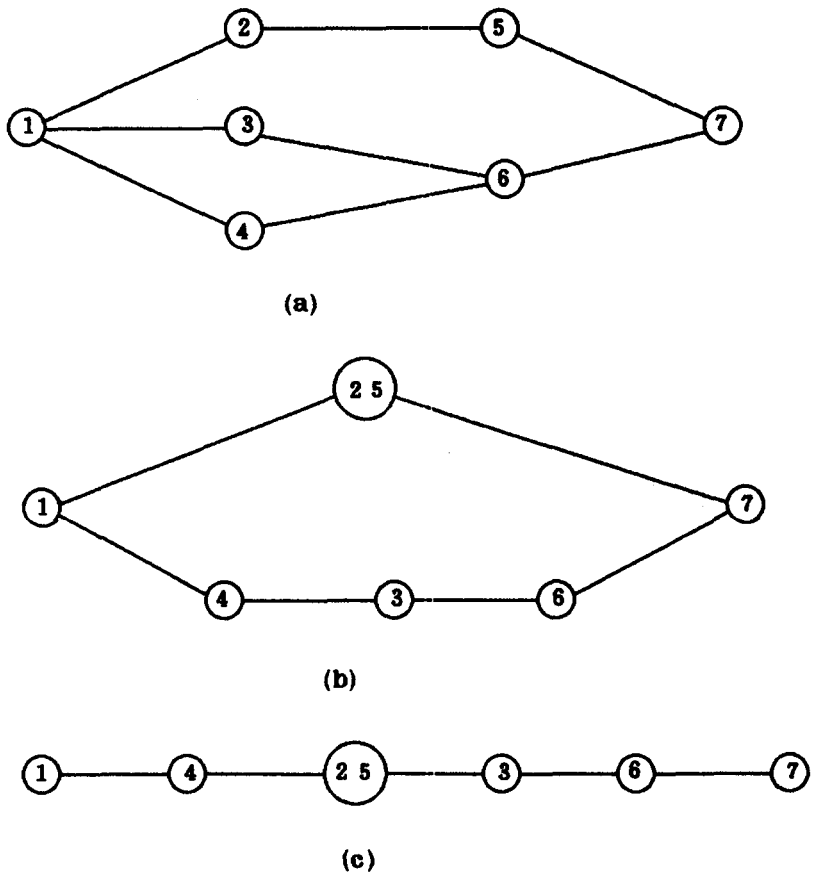


Fig. 5.3 Example 3.

6. Summary and Remarks

We proposed an algorithm which gives inter- and intra-group schedules minimizing the maximum completion times in a two-machine GT flow-shop. The series-parallel algorithm has been shown to require computation of order $O(g \log g)$, g; the number of nodes in R, if R is expressed in an adequate internal form [13]. Intra-group scheduling by the Johnson's rule needs computation of order $O(n \log n)$ where n is the job number in the group concerned. The proposed algorithm can be implemented by computation of order not greater than $O(n \log n)$ where n is the total number of jobs.

The main body of our discussion is composed of

(1) an optimal intra-group schedule of a job group is an optimal schedule of a problem restricted to the jobs in the group, whatever an inter-group schedule is (theorem 2),

(2) an optimal inter-group schedule under no precedence relation can be obtained by applying the generalized Johnson's rule for composite jobs corresponding to job groups (theorem 6),

(3) a composite job can be recognized as a natural generalization of an ordinary job (with a time lag) in a flow-shop, and

(4) a composite job scheduling problem conserves properties of a scheduling problem on ordinary jobs (sections 4 and 5).

(2) ~ (4) implies that inter-group and intra-group scheduling problems are essentially indifferent.

This fact suggests that a intra-group scheduling problem with type-AB jobs only (notice that all composite jobs in this paper are of type-AB) can be solved by replacing each job with a respective composite job and by using the series-parallel algorithm, even if

(1) each job has its setup times not included in the processing times,

(2) a series-parallel precedence relation is imposed among jobs.

Remember that Jackson [3] expanded the Johnson's result into a case where jobs of type-A, type-B, type-AB and type-BA are mixed. A similar extention is possible for the generalized Johnson's rule. See Sekiguchi [18] for details. Thus, our problem in the GT flow-shop can be solved as follows even when four types of job groups are mixed but there is no precedence relation among job groups of different types. First, find an optimal schedule of job groups of the same type under a precedence constraint among them. Then, combine them by Jackson's method. In the preceding paragraph, we considered only type-AB jobs, in introducing precedence relations among jobs in groups. But, it is now easy to see that an intra-group scheduling problem with type-

A, type-B type-AB and type-BA jobs (i.e. the type of the job group is not any-one of the standard four types) under a precedence constraint can also be solved as long as there is no precedence relation among jobs of different types. But, notice that a composite is not definable unless a job group is of type-AB or type-BA.

In this paper, we assumed that the buffer capacity between machines is infinite. Actually, the buffer capacity between machines may be finite so as to satisfy the assumption, if the capacity is large enough to keep jobs which machine B processes during transportation of overflowing jobs to/from a nearby warehouse. But, if such a warehouse is not available, we need to solve a two-machine flow shop problem under a finite buffer capacity, which is NP-complete with the exception of a case of capacity zero [15].

Consider that rescheduling jobs is necessary because of changes such as cancellation or addition of orders on the way of an implementation of a schedule. There is no difficulty to resolve the situation since the proposed algorithm solves efficiently the problems under arbitrary initial conditions. Determine the initial condition for remaining jobs (and job groups), and reschedule them by the proposed algorithm.

Thus, we see that our algorithm solves a variety of the two-machine minimum makespan flow shop problems. However, the algorithm does not resolve situations in which some jobs or some job groups must be started at some specified times or must be prohibited from being scheduled as the first (last) one.

Some cases of a three-machine flow-shop problem can be reduced into ones in a two-machine flow-shop (see [2], and also [13]). This has been shown principally based on the fact that the maximum completion time on the last machine of a permutation schedule of the three-machine flow-shop problem is equal to that of the same schedule of the respective two-machine flow-shop problem. It is easy to see that similar cases of a three-machine problem obtained by extending naturally our two-machine problem can be reduced to two-machine problems, if machine B is recessive. However, it is not clear that under what conditions this is true if machine B may be a bottleneck.

For a general case of the three-machine problems and for the problems with more than three machines, a branch-and-bound method has been effectively used. Likewise, for the GT flow shop problems with three or more machines, with the exception of some special cases of three machines, a branch-and-bound-like method will be the only one which can be used in practice. The proposed algorithm can be applied for lower bound calculations, and improves the efficiency of a branch-and-bound method [14].

Only permutation schedules were considered in this paper. In the minimum

makespan problem in the ordinary two-machine flow shop, the set of permutation schedules can happen not to include an optimal solution only when the following relation holds for some i and j [5].

$$D_i - \min(A_i, B_i) > D_j + \max(A_j, B_j) - \min(A_j, B_j) = D_j + |A_j - B_j|.$$

Therefore, there is an optimal schedule among permutation schedules for the intra-group scheduling problem only if there is no pair i and j satisfying the relation. When composite jobs are introduced, the effect of lag times appears intensively in Δ. Determine a composite job corresponding to a job with parameters A_i, B_i and D_i then $\Delta_i = \min(A_i, B_i) - D_i$ holds. The relation above is equivalent to

$$-\Delta_i > D_j + |A_j - B_j|.$$

An unsolved problem is to specify for general composite jobs the value of Δ which makes all permutation schedules nonoptimal.

Reference

[1] Bellman, R.: Mathematical Aspects of Scheduling Theory. *RAND Report p-651*, RAND Corporation, Santa Monica, April 11, 1955.

[2] Burns, F. and Rooker, J.: Three Stage Flow-Shops with Recessive Second Stage. *Operations Research*, Vol.26 (1978), 207-208.

[3] Jackson, J. R.: An Extension of Johnson's Results on Job Lot Scheduling. *Naval Research Logistics Quarterly*, Vol.3 (1956), 201-203.

[4] Johnson, S. M.: Optimal Two-and Three-Stage Production Schedules with Setup Times Included. *Naval Research Logistics Quarterly*, Vol.1 (1954), 61-68.

[5] Johnson, S. M.: Discussion: Sequencing n Jobs on Two Machines with Arbitrary Time Lags. *Management Science*, Vol.5 (1959), 299-303.

[6] Kurisu, T.: Two-Machine Scheduling under Required Precedence among Jobs. *Journal of the Operations Research Society of Japan*, Vol.19 (1976), 1-13.

[7] Kurisu, T.: Two-Machine Scheduling under Arbitrary Precedence Constraints. *Journal of the Operations Research Society of Japan*, Vol.20 (1977), 113-131.

[8] Lageweg, B. J., Lenstra, J. K. and Rinnooy Kan, A. H. G.: A General Bounding Scheme for the Permutation Flow-Shop Problem. *Operations Research*, Vol.26 (1978), 53-67.

[9] Lawler, E. L.: Sequencing Jobs to Minimize Total Weighted Completion Time Subject to Precedence Constraints. *Annals of Discrete Mathematics*, Vol.2 (1978), 75-90.

[10] Maggu, P. L., Das, G. and Kumar, R.: On Equivalent Job for Job-Block in 2 × n Sequencing Problem with Transportation-Times. *Journal of the Operations Research Society of Japan*, Vol.24 (1981), 136-146.

[11] Mitten, L. G.: Sequencing n Jobs on Two Machines with Arbitrary Time Lags. *Management Science*, Vol.5 (1959), 293-298.

[12] Mitten, L. G.: A Scheduling Problem. *The Journal of Industrial Engineering*, Vol.10 (1959), 131-135.

[13] Monma, C. L.: The Two-Machine Maximum Flow Time Problem with Series-Parallel Precedence Constraints: An Algorithm and Extensions. *Operations Reserach*, Vol.27 (1979), 792-798.

[14] Nakanishi, Y.: On the Efficiency of Revised Lower Bound in a Group Scheduling Problem (in Japanese). Manuscript (1983, 2).

[15] Papadimitriou, C. H. and Kanellakis, P. C.: Flowshop Scheduling with Limited Temporary Storage. *Journal of the Association for Computing Machinery*, Vol.27 (1980), 533-549.

[16] Sekiguchi, Y., Koyama, S. and Miura, R.: Flow Shop and Analysis of the Schedules. *Transaction of the Society of Instrument and Control Engineers*, Vol.8 (1972), 40-48.

[17] Sekiguchi, Y.: On Lower Bounds for Flow-Shop Scheduling Problems. *Hokudai Keizaigaku Kenkyu*, Vol.29 (1979), 139-161.

[18] Sekiguchi, Y.: Inter- and Intra-Group-of-Jobs Schedule for Minimizing Makespan in a 2-Machine GT Shop. *The 8th IFAC International Congress Preprints*, Vol.XIV (1981), XIV 164-169.

[19] Sidney, J. B.: The Two-Machine Maximum Flow Time Problem with Series-Parallel Precedence Relations. *Operations Research*, Vol.27 (1979), 782-791.

Yasuki SEKIGUCHI: Faculty of
Economics, Hokkaido University,
Kita-Ku Kita 9 Nishi 7
Sapporo, 060, Japan.