OPTIMAL SCHEDULES UNDER SERIES-PARALLEL PRECEDENCE CONSTRAINTS

YASUKI SEKIGUCHI
Associate Professor
Faculty of Economics, Hokkaido University

ABSTRACT

A class of permutation scheduling problems are discussed. Suppose that a permutation scheduling problem without precedence constraints is given a solution as a dominance relation, i. e. an optimal solution is obtained by arranging jobs according to the dominance relation. Then, a sufficient condition for the same problem imposed a series-parallel precedence constraint to be solved by a series-parallel algorithm of O ($n \log n$), n : the number of jobs, is clarified. Five example problems are shown to satisfy this sufficient condition. Two of them are a single machine scheduling problem minimizing total weighted completion time and a two-machine flow-shop scheduling problem minimizing the maximum completion time, both have been given series-parallel algorithms elsewhere. The other three are a problem minimizing the maximum cumulative cost, a single machine scheduling problem minimizing the maximum lateness and a single machine scheduling problem minimizing total discounted cost. These three problems have not been known to have series-parallel algorithms.

1. INTRODUCTION

Consider problems where optimal permutations under precedence constraints are sought. We will discuss below series-parallel relations as the precedence constraints.

Lawler [6] defined series-parallel networks in a recursive way, in developing the work of Sidney [11] on a single machine scheduling problem minimizing total weighted completion time. His definition is constructive. Start with trivial single vertex networks, repeat their series composition and parallel composition, and a resultant network is called a series-parallel network.

Along with Lawler's result, Sidney [12] utilized series-parallel networks in developing the results of Kurisu [4]. His definition of series-parallel networks is also recursive, but reductive rather than constructive (see 3). The scheduling problem considered was a two-machine flow shop problem minimizing makespan. Monma [8] developed Sidney's results further and

obtained an algorithm whose computational requirement is O $(n \log n)$, n: the number of jobs.

Notice that these single and two machine scheduling problems have little similarity to each other and logics to their solution algorithms are different. But, the two solution algorithms have basically the same structure. The basic algorithmic structure here means that the algorithm consists of a repetitive operation that exchanges a parallel subnetwork of the original precedence network into a chain, constructed according to the optimal solution of a restricted scheduling problem on the subnetwork. Algorithms with this basic structure are called series-parallel algorithms.

A set of conditions under which the series-parallel algorithms work well will be clarified. The problems cited will be shown to satisfy these conditions. Besides, as examples of application of the theory, some other problems will be shown to have the same properties.

2. PARALLEL CHAIN ALGORITHM

Let S be a set of objects with some specific attributes. A subset of S is denoted as N. Let n be the number of elements in N. Suppose that the elements of N is numbered from 1 to n, and N also represents the set of this numbers, i.e. $N=\{1, 2, \cdots, n\}$. Thus, a subset K of N is a set of elements in S as well as the set of numbers assigned to the elements in N. Each element in N, and also in S, is called a job. A schedule is a permutation of jobs. $\sigma_1 \cdot \sigma_2$ denotes a schedule obtained by concatenating σ_2 after σ_1. Here, the set of jobs in σ_1 and the set of jobs in σ_2 are assumed to be disjoint. A function f is a mapping from a schedule σ of an arbitrary length to a real number.

Denote as P a problem where a feasible schedule minimizing $f(\sigma)$ is searched for, when there is no explicit precedence constraint among jobs.

[assumption 1] (existency of a dominance relation $\leqq$) A total order is definable on S, and for an arbitrary pair of jobs, i and j, whether $i \leqq j$ or $j \leqq i$ can be determined from the attributes of these two elements.

Here, the total order on S is one that satisfied reflexity, anti-symmetricity and transitivity and that one of $i \leqq j$ and $j \leqq i$ holds for every pair of jobs in S. Therefore, given a finite subset N of S, it is possible to make a schedule by arranging all elements of N according to the relation $\leqq$.

[assumption 2] (optimality of the dominance relation $\leqq$) The schedule of N determined according to the dominance relation $\leqq$ is optimal for P.

The assumption 3 on f requires that any schedule is improved as a whole by improving its subschedule. This property is named the series network decomposition (SND) property by Sidney [13].

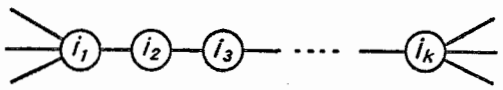
[assumption 3] (SND property) Suppose $\sigma_2 \neq \phi$ in an arbitrary schedule

$\sigma_1 \cdot \sigma_2 \cdot \sigma_3$. Let σ'_2 be a schedule of jobs in σ_2 different from σ_2. Then, $f(\sigma_2) \leqq f(\sigma'_2)$ implies $f(\sigma_1 \cdot \sigma_2 \cdot \sigma_3) \leqq f(\sigma_1 \cdot \sigma'_2 \cdot \sigma_3)$.

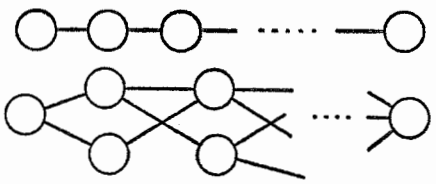
[assumption 4] (membership of composite jobs to S) It is possible to define a composite job $J_\sigma \in S$ for an arbitrary (sub) schedule σ, and the composite job satisfies $f(\sigma_1 \cdot \sigma \cdot \sigma_2) = f(\sigma_1 \cdot J_\sigma \cdot \sigma_2) + \Delta_\sigma$, where Δ_σ is a constant determined by σ and independent of σ_1 and σ_2.

The assumption 4 requires that, when a schedule σ of a subset of N is fixed, an optimal schedule of N (including σ as its consecutive subschedule) is obtained by solving a reduced problem of P where the set of jobs in σ is exchanged into J_σ.

Define $G=(N, E)$ as a network with a vertex set N and arc set E. An arc $i \to j$ implies that job i must precede job j, i. e. job i must be scheduled before job j. A *chain* is a sequence $(i_1, i_2, \cdots, i_k)$ such that $i_1 \to i_2 \to \cdots \to i_k$ and every j which is not in the sequence either succeed or precede all jobs in the sequence (Fig. 1 (a)). If two networks are disconnected, they are said to be *parallel* (Fig. 1 (b)). Notice that any arc $i \to j$ is a chain if both out-degree of i and indegree of j are one.



(a) A chain.



(b) Two parallel networks.

Fig. 1. A chain and parallel networks.

A problem P with a precedence relation constraint G is denoted as (P, G).

[theorem 1] Suppose that a problem (P, G) satisfies the assumptions 1$\sim$4, where G is an arbitrary network. If $i \geqq j$ holds for an arc $i \to j$ which is a chain of G, then there is an optimal schedule of (P, G) such that job i is scheduled directly before job j.

[proof] Let $\sigma_1 \cdot i \cdot \sigma \cdot j \cdot \sigma_2$ be an optimal schedule of (P, G) such that $\sigma \neq \phi$. $i \to j$ is a chain and all jobs in σ have no precedence relation with i and j,

because otherwise the schedule can not be feasible. From assumption 4,

$$f(\sigma_1 \cdot i \cdot \sigma \cdot j \cdot \sigma_2) = f(\sigma_1 \cdot i \cdot J_\sigma \cdot j \cdot \sigma_2) + \Delta_\sigma .$$

If $J_\sigma \leqq j$, then $J_\sigma \leqq j \leqq i$ must be true by virtue of assumption 1 (i. e. transitivity). And, by assumption 2, we get $f(J_\sigma \cdot i) \leqq f(i \cdot J_\sigma)$. Besides, assumption 3 implies $f(\sigma_1 \cdot J_\sigma \cdot i \cdot j \cdot \sigma_2) \leqq f(\sigma_1 \cdot i \cdot J_\sigma \cdot j \cdot \sigma_2)$. Combining these results, it is easy to see

$$\begin{aligned} f(\sigma_1 \cdot \sigma \cdot i \cdot j \cdot \sigma_2) &= f(\sigma_1 \cdot J_\sigma \cdot i \cdot j \cdot \sigma_2) + \Delta_\sigma \\ &\leqq f(\sigma_1 \cdot i \cdot J_\sigma \cdot j \cdot \sigma_2) + \Delta_\sigma \\ &= f(\sigma_1 \cdot i \cdot \sigma \cdot j \cdot \sigma_2) \end{aligned}$$

If $J_\sigma \geqq j$, it will be shown through a similar process that

$$\begin{aligned} f(\sigma_1 \cdot i \cdot j \cdot \sigma \cdot \sigma_2) &= f(\sigma_1 \cdot i \cdot j \cdot J_\sigma \cdot \sigma_2) + \Delta_\sigma \\ &\leqq f(\sigma_1 \cdot i \cdot \sigma \cdot \cdot j \cdot \sigma_2) . \end{aligned}$$

Assumption 5 and theorem 2 below are not referred to in the succeeding development, but they are useful for a smooth implementation of the parallel chain algorithm defined afterward.

[assumption 5] (conservation of dominance relations) Let $\sigma = i \cdot j$. If $i \geqq j$, then $i \geqq J_\sigma \geqq j$.

[theorem 2] Suppose that a problem (P, G) satisfies assumptions 1$\sim$5. Suppose also that G is an arbitrary network which includes a chain $i_1 \rightarrow i_2 \rightarrow \cdots \rightarrow i_k$ as a subnetwork. If $i_1 \geqq i_2 \geqq \cdots \geqq i_k$ for this chain, then there is an optimal schedule of (P, G) that includes a sequence $\sigma = i_1 \cdot i_2 \cdots i_k$ as a consecutive subschedule.

[proof] Exchange $i_1 \cdot i_2$ into J_{12} by applying theorem 1 on $i_1 \rightarrow i_2$. Then, $J_{12} \geqq i_3$ holds for $J_{12} \rightarrow i_3$ by assumption 5, and theorem 1 is again applicable. The proof is established by applying theorem 1 and assumption 5 repeatedly.

Thus, we can deduce the following algorithm.

[parallel chain algorithm]

(1) For each chain, repeatedly exchange into a composite job two jobs of an arc whose direction and the dominance relation between them are in conflict. At termination of this repetition, the original parallel chain has been reduced to one that has no conflict between arcs and dominance relations.

(2) Make a schedule of (composite) jobs in the resultant parallel chain by arranging them according to dominance relations. Substitute subschedules of original jobs corresponding to composite jobs.

[theorem 3] Suppose that the precedence relation G in a problem (P, G) satisfying assumptions 1$\sim$4 is a parallel chain. Then, a schedule given by

the parallel chain algorithm is optimal.

[proof] The first step of the parallel chain algorithm is verified by theorem 1. The parallel chain given by this first step has no conflict between arcs and dominance relations, and the schedule given by the second step naturally satisfies the precedence constraints.

3. SERIES-PARALLEL ALGORITHM

It is shown here that optimal schedules under further general precedence constraints, which are constructed by combining parallel chains in series and/or in parallel, can be obtained by applying the parallel-chain algorithm repeatedly.

A *parallel chain subnetwork* in an arbitrary network is one such that (i) it consists of a set of parallel chains and (ii) the first (last) vertex in each chain has the same set of preceding (succeeding) vertices as the others do.

A network is called *a series-parallel network if it can be reduced into a single chain by applying repeatedly the following operation; exchange a parallel chain subnetwork with a chain (corresponding to a feasible schedule of jobs in the subnetwork).*

As stated later, the series-parallel algorithm utilizes repetitively the parallel chain algorithm and is a kind of procedure to reduce a series-parallel network into a single chain. The readers can probably understand intuitively the family of problems (P, G) solvable by the series parallel algorithm and its validity. We will discuss these issues along with Sidney's [12] theory.

[theorem 4] Suppose that G in a problem (P, G) satisfying assumptions $1\sim4$ consists of r subnetworks which are $r-1$ parallel chains and an arbitrary subnetwork (Fig. 2). Denote as G^* the subnetwork of $r-1$ parallel chains. Let (P^*, G^*) be the reduced problem of (P, G) restricted on the jobs

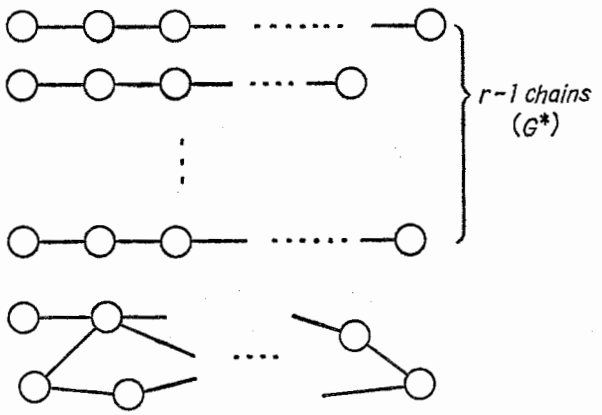


Fig. 2. The r parallel networks in theorem 4.

in G^*. Let σ^* be the optimal schedule of (P^*, G^*) obtained through the parallel chain algorithm. Then, there are optimal schedules of (P, G) which include σ^* as a (not necessarily consecutive) subschedule.

[proof] Let γ be an arbitrary optimal schedule of (P, G). Let α be a subschedule of γ eliminated jobs in G^*, therefore in σ^*. α is a feasible schedule of jobs in the r-th subnetwork. Consider a problem (P, G_α), where G_α is a parallel chain consisting of the original $r-1$ chains and a chain corresponding to α. Notice that r is an optimal schedule of (P, G_α), and therefore that every optimal schedule of (P, G_α) is also optimal for (P, G). Thus, the proof is completed by showing that there is at least one optimal schedule of (P, G_α) which includes σ^* as a subschedule. This can be done as follows. The first step of the parallel chain algorithm is performed on each chain independently. The result of this step is not affected by the other chains parallel to it. Thus, the result of the first step applied for the original $r-1$ parallel chains in (P, G_α) must be identical to one for (P^*, G^*). Notice also that dominance relations among (composite) jobs in the $r-1$ parallel chains in the resultant network of (P, G_α) must be identical to ones among those in the resultant network of (P^*, G^*) (cf. assumption 1). This implies that σ^* must be included as a subschedule in the schedule obtained by the second step of the parallel chain algorithm applied for the resultant network of (P, G_α).

Based on theorem 4, the following theorem, which is essential in constructing the series-parallel algorithm and in proving its validity, is established.

[theorem 5] Suppose that G in (P, G) satisfying assumptions $1\sim4$ contains a parallel chain subnetwork G^*. A problem (P^*, G^*) is defined by restricting (P, G) on G^*, and it is a problem on a parallel chain network. Determine an optimal schedule of this problem by the parallel chain algorithm and denote it as σ^*. Then, the problem (P, G) has at least one optimal schedule which contains σ^* as a (not necessarily consecutive) subschedule.

[proof] Let $\sigma=\sigma_1\cdot i\cdot\sigma_2\cdot j\cdot\sigma_3$ be an arbitrary optimal schedule of (P, G). Here, $i(j)$ is the first (last) job in σ included in G^*. σ_2 contains all jobs except i and j in G^* and a set K of jobs not included in G^*. Notice that any job in K does not have a precedence relation with jobs in G^*, because G^* is a parallel chain subnetwork. Let G^0 be the subnetwork of G induced by i, j and jobs in σ_2. Then G^0 consists of a parallel chain subnetwork G^* and another one parallel to it. It satisfies the condition of theorem 4. Therefore, the restricted problem (P^0, G^0) of (P, G) has an optimal schedule σ^0 containing σ^* as a subschedule. This implies $f(\sigma^0)\leqq f(i\cdot\sigma_2\cdot j)$. And by assumtion 3, $f(\sigma_1\cdot\sigma^0\cdot\sigma_3)\leqq f(\sigma_1\cdot i\cdot\sigma_2\cdot j\cdot\sigma_3)$. $\sigma_1\cdot\sigma^0\cdot\sigma_3$ is proved being an optimal schedule of (P, G) which is asserted by the theorem.

The series-parallel algorithm is defined as follows.

[series-parallel algorithm]

(1) Exchange a parallel chain subnetwork of G with a chain corresponding to a subschedule obtained by applying the parallel chain algorithm for the parallel chain subnetwork.

(2) Repeat (1) until the resultant network becomes a single chain which has no conflict between dominance relations and precedence relations.

[theorem 6] Suppose that G in a problem (P, G) satisfying assumption $1\sim4$ is a series-parallel network. Then, the series-parallel algorithm gives an optimal schedule of (P, G).

[proof] Theorem 5 guarantees that optimal schedules of the problem obtained as a result of (1) are optimal for (P, G), if composite jobs in them are exchanged into corresponding subschedules of original jobs. Moreover, the condition for termination required in (2) can surely be reached because of the definition of series-parallel networks.

The complexity of the series-parallel algorithm is shown to be $0\ (n \log n)$ under an adequate internal representation of networks (Lawler [6]).

Observing carefully theorem 5 or its base, i. e. theorem 4 and the proof, we will understand the following.

An essential point in the statement of theorem 4 is that an algorithm is known to solve a problem given by restricting an original problem (P, G) on a subnetwork of G with some specific properties and to be independent of the other part of G. Call this kind of algorithm a *X-algorithm* and the subnetwork a *X-subnetwork.* A family of networks is *X-reducible* if any one of them contains an X-subnetwork and it has at least one optimal schedule containing the subschedule given by the X-algorithm. In this sense, G in theorem 4 is parallel-chain-reducible.

Let G be a general network with some X-subnetworks. $i(j)$ is the first (last) job in an X-subnetwork of G in an optimal schedule σ of a problem (P, G). Then, notice that an essential thing for establishing theorem 5 is: a subnetwork of G induced by jobs in a subschedule $i\cdot\sigma_2\cdot j$ of σ is X-reducible. $i\cdot\sigma_2\cdot j$ is determined independently of the X-algorithm and whether the subnetwork is X-reducible or not depends only on the definition of X-subnetworks.

Suppose that we could find an elementary network (X-network), other than parallel chains, which is solvable and gives X-reducibility. Then, it will be possible to determine a family of networks, so to say X-recursive networks, with a property that scheduling problems defined on them can be solved by a recursive application of a X-algorithm.

4. APPLICATIONS

The effectiveness of the preceding result will be confirmed here by applying theorem 6 for some specific examples.

4.1 A single machine scheduling problem minimizing total weighted completion time.

Let S be a set of pairs (p, w) of a positive processing time p and a nonnegative weight w. (p_i, w_i) is an element of S corresponding to an element i in $N=\{1, 2, \cdots, n\}$. The i-th job of an arbitrary schedule σ of N is denoted as $\langle i\rangle$. Define

$$f(\sigma)=\sum_{i=1}^{n}\{w_{\langle i\rangle}\sum_{j=1}^{i}p_{\langle j\rangle}\},$$

and

$$C_{\langle i\rangle}=\sum_{j=1}^{i}p_{\langle j\rangle}.$$

$C_{\langle i\rangle}$ is the completion time of job $\langle i\rangle$ and f is the weighted total of completion times under σ. A schedule minimizing f is given by arranging jobs according to a dominance relation (Smith [14]),

$$i\leqq j \quad\leftrightarrow\quad -w_i/p_i\leqq -w_j/p_j.$$

This relation $\leqq$ is evidently a total order, and assumptions 1 and 2 are satisfied. Satisfication of assumption 3 can be shown as follows. Let for simplicity, but without loss of generality, $\sigma_1=1\cdot 2\cdots k$, $\sigma_2=k+1\cdot k+2\cdots l-1$ and $\sigma_3=l\cdot l+1\cdots n$. $\sigma'_2=\langle k+1\rangle\cdot\langle k+2\rangle\cdots\langle l-1\rangle$ is another schedule of jobs in σ_2. Suppose that σ'_2 satisfies,

$$\begin{aligned}f(\sigma_2)&=\sum_{i=k+1}^{l-1}\{w_i\sum_{j=k+1}^{i}p_j\}\\&\leqq\sum_{i=k+1}^{l-1}\{w_{\langle i\rangle}\sum_{j=k+1}^{i}p_{\langle j\rangle}\}=f(\sigma'_2).\end{aligned}$$

Notice that,

$$\begin{aligned}f(\sigma_1\cdot\sigma_2\cdot\sigma_3)&=\sum_{i=1}^{k}\{w_i\sum_{j=1}^{i}p_j\}+\sum_{i=k+1}^{l-1}\{w_i\sum_{j=1}^{i}p_j\}\\&\quad+\sum_{i=1}^{n}\{w_i\sum_{j=1}^{i}p_j\}\\&=\sum_{i=1}^{k}\{w_i\sum_{j=1}^{i}p_j\}+\sum_{i=k+1}^{l-1}\{w_i(C_k+\sum_{j=k+1}^{i}p_j)\}\\&\quad+\sum_{i=1}^{n}\{w_i(C_k+\sum_{j=k+1}^{l-1}p_j+\sum_{j=1}^{i}p_j)\}.\end{aligned}$$

Substituting $f(\sigma_2)\leqq f('_2)$ and

$$\sum_{i=k+1}^{l-1}w_i=\sum_{i=k+1}^{l-1}w_{\langle i\rangle},\qquad \sum_{j=k+1}^{l-1}p_j=\sum_{j=k+1}^{l-1}p_{\langle j\rangle}$$

into the formula above, we will get

$$\begin{aligned}f(\sigma_1\cdot\sigma_2\cdot\sigma_3)&\leqq\sum_{i=1}^{k}\{w_i\sum_{j=1}^{i}p_j\}\\&\quad+\sum_{i=k+1}^{l-1}w_{\langle i\rangle}(C_k+\sum_{j=j+1}^{i}p_{\langle j\rangle})\}\\&\quad+\sum_{i=1}^{n}\{w_i(C_k+\sum_{j=k+1}^{l-1}p_{\langle j\rangle}+\sum_{j=1}^{i}p_j)\}\end{aligned}$$

$$= f(\sigma_1 \cdot \sigma'_2 \cdot \sigma_3) .$$

The last thing to be shown is satisfication of assumption 4. Define a composite job $J_\sigma=(p, w)$ corresponding to a subschedule σ as follows.

$$p = \sum_{j=k+1}^{l-1} p_{\langle j \rangle}, \qquad w = \sum_{j=k+1}^{l-1} w_{\langle j \rangle} .$$

where it is assumed that σ is preceded by σ_1 of length k and $\sigma=\langle k+1\rangle \cdots \langle l-1\rangle$. Besides, let σ_2 be a schedule succeeding σ of length $n-l+1$. Then, utilizing the formula for $f(\sigma_1 \cdot \sigma_2 \cdot \sigma_3)$ above, we will get,

$$\begin{aligned} f(\sigma_1 \cdot \sigma \cdot \sigma_2) = & \sum_{i=1}^{k} \{w_{\langle i \rangle} \sum_{j=1}^{i} p_{\langle j \rangle}\} + C_k \sum_{i=k+1}^{l-1} w_{\langle i \rangle} \\ & + \sum_{i=k+1}^{l-1} \{w_{\langle i \rangle} \sum_{j=j+1}^{i} p_{\langle j \rangle}\} \\ & + \sum_{i=1}^{n} \{w_{\langle i \rangle}(C_k + p + \sum_{j=1}^{i} p_{\langle j \rangle})\} \end{aligned}$$

The third term of the righthand side can be restated as follows.

$$\begin{aligned} \text{the 3rd term} = & w_{\langle k+1 \rangle} p_{\langle k+1 \rangle} \\ & + w_{\langle k+2 \rangle} p_{\langle k+2 \rangle} + w_{\langle k+2 \rangle} p_{\langle k+1 \rangle} \\ & \vdots \\ & + w_{\langle l-1 \rangle} p_{\langle l+1 \rangle} + \cdots + w_{\langle l-1 \rangle} p_{\langle k+1 \rangle} \\ = & wp - \sum_{i=k+1}^{l-2} \{w_{\langle i \rangle} \sum_{k=i+1}^{l-1} p_{\langle j \rangle}\} . \end{aligned}$$

Let the last term above be denoted as $-\Delta_\sigma$, then

$$\begin{aligned} f(\sigma_1 \cdot \sigma \cdot \sigma_2) = & \sum_{i=1}^{k} \{w_{\langle i \rangle} \sum_{j=1}^{i} p_{\langle j \rangle}\} + C_k w + wp \\ & + \sum_{i=1}^{n} \{w_{\langle i \rangle}(C_k + p + \sum_{j=1}^{i} p_{\langle i \rangle})\} + \Delta_\sigma \\ = & f(\sigma_1 \cdot J_\sigma \cdot \sigma_2) + \Delta_\sigma . \end{aligned}$$

It is easy to see that assumption 5 is satisfied because $-w_1/p_1 \geqq -w_2/p_2$ implies $-w_1/p_1 \geqq -(w_1+w_2)/(p_1+p_2) \geqq -w_2/p_2$.

4.2 A two machine flow-shop scheduling problem minimizing the maximum completion time.

S is a set of two-dimensional vectors (p, q), where p and q can be negative. A positive p and a negative q represent processing times of length $|p|$ and $|q|$, respectively, on the first machine, where as a negative p and a positive q do those on the second machine. An element i of N corresponds to an element (p_i, q_i) of S. $T_1(\sigma)$ and $T_2(\sigma)$ denote completion times on the first and second machines, respectively, of a schedule σ of (a subset of) N. The completion times are calculated according to a recurrence relation,

$$\begin{aligned} & T_1(\sigma \cdot i) = T_1(\sigma) + \max\{p_i, 0\} - \min\{0, q_i\} \\ \text{(A)} \qquad & T_2(\sigma \cdot i) = \max \begin{bmatrix} T_1(\sigma) + \max\{p_i, 0\} \\ T_2(\sigma) - \min\{0, p_i\} \end{bmatrix} + \max\{q_i, 0\} \end{aligned}$$

starting from an initial condition,

$$T_1(\phi)=0\,, \qquad T_2(\phi)=t_0\,, \qquad \text{where } t_0 \text{ is an arbitrary real.}$$

Then, the problem P here is to find a schedule of N minimizing $f(\sigma)=T_2(\sigma)$. If p and q are nonnegative, this problem is identical to the classic Johnson's problem [3]. Please refer to Sekiguchi [9, 10] for the effects and the origin of allowing negative p and q. We note here only that a variety of two machine flow-shop problems minimizing the total completion time can be understood as special cases of P.

Define a dominance relation as follows.

$$\begin{aligned} i\leqq j \quad \leftrightarrow \quad & p_i\leqq q_i\,, \quad p_j\leqq q_j \text{ and } p_i\leqq p_j\,, \\ & \text{or} \quad p_i\leqq q_i \text{ and } p_j\leqq q_j\,, \\ & \text{or} \quad p_i\geqq q_i\,, \quad p_j\geqq q_j \text{ and } q_i\geqq q_j\,. \end{aligned}$$

This dominance relation is transitive under a slight restriction, and a total order. An optimal schedule is obtained by arranging jobs according to it [10]. That is, assumptions 1 and 2 are satisfied. It is easy to see that assumption 3 is satisfied, if we use (1) $i\leqq j$ implies $T_1(i\cdot j)=T_1(j\cdot i)$ and $T_2(i\cdot j)\leqq T_2(j\cdot i)$ under an arbitrary initial condition (theorem 5 in [10]), and (2) $T_1(\sigma\cdot i)$ and $T_2(\sigma\cdot i)$ are non-decreasing for $T_1(\sigma)$ and $T_2(\sigma)$.

A composite job $J_\sigma=(p, q)$ corresponding to a subschedule $\sigma=(i\cdot j)$ is defined by the following formulas.

$$\begin{aligned} p &= p_i+\max\{p_j-q_i, 0\} \\ q &= q_j+\max\{q_i-p_j, 0\}\,. \end{aligned}$$

By a recursive calculation of (A), it will be seen

$$\text{(B)} \qquad T_1(\sigma)=T_1(J_\sigma)+\Delta_\sigma\,, \qquad T_2(\sigma)=T_2(J_\sigma)+\Delta_\sigma\,,$$

where Δ_σ is a constant dependent only on p_i, q_i, p_j and q_j. A composite job for a subschedule of length three or more is determined by repeating determinations of composite jobs of subschedules of length two. For example, if $\sigma=i\cdot j\cdot k$, first determine J_σ for $\alpha=i\cdot j$, then J_σ for a suppositional schedule $J_\sigma\cdot k$. It is not difficult to see that assumptions 4 and 5 are satisfied, see [10] for detail.

4. 3 A problem minimizing the maximum cumulative cost.

This is a generalization of a problem which has been shown by Abdel-Wahab and Kameda [1] to have an exact solution of computational requirement of order not greater than O (n^2) (Sidney [13]). This problem is shown here having a solution of order O ($n \log n$), i. e., a series-parallel algorithm, by proving that assumptions 1$\sim$4 are satisfied.

S is a set of two dimensional vectors (c, m). An element (c_i, m_i) of S corresponds to an element i of N. Let $f(\phi)=c_0$ and $c_{\langle 0\rangle}=c_0$, then our problem P is to find a schedule minimizing

$$f(\sigma)=\max_{1\leqq i\leqq n}\{CC_{\langle i\rangle}\}=\max_{1\leqq i\leqq n}\{\textstyle\sum_{j=0}^{i-1}c_{\langle j\rangle}+m_{\langle i\rangle}\}\ .$$

Here, c_i and m_i are arbitrary reals.

Define a dominance relation as follows.

$$\begin{aligned} i\leqq j \ \leftrightarrow\ & c_i\leqq 0\,,\quad c_j\leqq 0 \text{ and } m_i\leqq m_j\,,\\ \text{or}\ & c_i\leqq 0 \text{ and } c_j\geqq 0\,,\\ \text{or}\ & c_i\geqq 0\,,\quad c_j\geqq 0 \text{ and } m_i-c_i\geqq m_j-c_j\,. \end{aligned}$$

It is easy to see that this dominance relation is transitive under a slight condition and a total order. Thus, assumption 1 is satisfied.

Suppose $i\leqq j$ for an arbitrary schedule $\sigma_1\cdot j\cdot i\cdot\sigma_2$. Let p be the length of σ_1, then

$$f(\sigma_1\cdot j\cdot i\cdot\sigma_2)=\max\left\{\begin{array}{l}\max\limits_{1\leqq k\leqq p}\{\sum_{l=0}^{k-1}c_{\langle l\rangle}+m_{\langle k\rangle}\}\,,\\ \sum_{l=0}^{p}c_{\langle l\rangle}+\max\{m_j,\ c_j+m_i\}\,,\\ \sum_{l=0}^{p}c_{\langle l\rangle}+c_j+c_i+\max\limits_{p+3\leqq k\leqq n}\{\sum_{l=p+3}^{k-1}c_{\langle l\rangle}+m_{\langle k\rangle}\}\end{array}\right\}$$

On the other hand,

$$f(\sigma_1\cdot i\cdot j\cdot\sigma_2)=\max\left\{\begin{array}{l}\max\limits_{1\leqq k\leqq p}\{\sum_{l=0}^{k-1}c_{\langle l\rangle}+m_{\langle k\rangle}\}\,,\\ \sum_{l=0}^{p}c_{\langle l\rangle}+\max\{m_i,\ c_i+m_j\}\,,\\ \sum_{l=0}^{p}c_{\langle l\rangle}+c_i+c_j+\max\limits_{p+3\leqq k\leqq n}\{\sum_{l=p+3}^{k-1}c_{\langle l\rangle}+m_{\langle k\rangle}\}\end{array}\right\}$$

Notice that

$$\begin{aligned} &c_i\leqq 0\,,\quad c_j\leqq 0 \text{ and } m_i\leqq m_j \text{ implies } m_j\geqq m_i \text{ and } m_j\geqq c_i+m_j\,,\\ &c_i\leqq 0 \text{ and } c_j\geqq 0 \text{ implies } m_j\geqq c_i+m_j \text{ and } c_j+m_i\geqq m_i\\ &c_i\geqq 0\,,\quad c_j\geqq 0 \text{ and } m_i-c_i\geqq m_j-c_j\\ &\qquad\text{implies } c_j+m_i\geqq m_j+c_i \text{ and } c_j+m_i\geqq m_i\,. \end{aligned}$$

Thus, the second term of $f(\sigma_1\cdot i\cdot j\cdot\sigma_2)$ is not greater than that of $f(\sigma_1\cdot j\cdot i\cdot\sigma_2)$, and $f(\sigma_1\cdot i\cdot j\cdot\sigma_2)\leqq f(\sigma_1\cdot j\cdot i\cdot\sigma_2)$. This implies that an arbitrary optimal schedule can be changed with no increase of the objective function value into a schedule according to the dominance relation by exchanging repeatedly two succeeding jobs which do not agree with the dominance relation.

Define σ_1, σ_2, σ_3 and σ'_2 as in 4. 1. Then,

$$f(\sigma_1\cdot\sigma_2\cdot\sigma_3)=\max\left\{\begin{array}{l}\max\limits_{1\leqq i\leqq k}\{\sum_{j=0}^{i-1}c_j+m_i\}\,,\\ \sum_{j=0}^{k}c_j+\max\limits_{k+1\leqq i\leqq l-1}\{\sum_{j=k+1}^{i-1}c_j+m_i\}\,,\\ \sum_{j=0}^{l-1}c_j+\max\limits_{l\leqq i\leqq n}\{\sum_{j=1}^{i-1}c_j+m_i\}\end{array}\right\}$$

$$f(\sigma_1\cdot\sigma'_2\cdot\sigma_3)=\max\left\{\begin{array}{l}\max\limits_{1\leqq i\leqq k}\{\sum_{j=0}^{i-1}c_j+m_i\}\ ,\\ \sum_{j=0}^{k}c_j+\max\limits_{k+1\leqq i\leqq l-1}\{\sum_{j=k+1}^{i-1}c_{\langle j\rangle}+m_{\langle i\rangle}\}\ ,\\ \sum_{j=0}^{k}c_j+\sum_{j=k+1}^{l-1}c_{\langle j\rangle}+\max\limits_{l\leqq i\leqq n}\{\sum_{j=1}^{i-1}c_j+m_i\}\end{array}\right\}$$

Assume here

$$\begin{aligned}f(\sigma_2)&=c_0+\max_{k+1\leqq i\leqq l-1}\{\sum_{j=k+1}^{i-1}c_j+m_i\}\\&\leqq c_0+\max_{k+1\leqq i\leqq l-1}\{\sum_{j=k+1}^{i-1}c_{\langle j\rangle}+m_{\langle i\rangle}\}=f(\sigma'_2)\ .\end{aligned}$$

then, it is easy to show

$$f(\sigma_1\cdot\sigma_2\cdot\sigma_3)\leqq f(\sigma_1\cdot\sigma'_2\cdot\sigma_3)\ ,$$

i. e., assumption 3 is satisfied.

Remember that σ_2 above is an arbitrary subschedule. Define a composite job $J_{\sigma 2}=(c, m)$ corresponding to σ_2 as follows.

$$\begin{aligned}&c=\sum_{j=k+1}^{l-1}c_j\ ,\\&m=\max_{k+1\leqq i\leqq l-1}\{\sum_{j=k+1}^{i-1}c_j+m_i\}\ .\end{aligned}$$

Then, the equation below holds and assumption 4 is satisfied.

$$\begin{aligned}f(\sigma_1\cdot\sigma_2\cdot\sigma_3)&=\max\left\{\begin{array}{l}\max\limits_{1\leqq i\leqq k}\{\sum_{j=0}^{i-1}c_j+m_i\}\ ,\\ \sum_{j=0}^{k}c_j+m\ ,\\ \sum_{j=0}^{k}c_j+c+\max\limits_{l\leqq i\leqq n}\{\sum_{j=1}^{i-1}c_j+m_i\}\end{array}\right\}\\&=f(\sigma_1\cdot J_{\sigma 2}\cdot\sigma_3)\ .\end{aligned}$$

In order to prove satisfication of assumption 5, let $\sigma=i\cdot j$ and $i\geqq j$. Notice that $c=c_i+c_j$ and $m=\max\{m_i, c_i+m_j\}$ for J_σ. If $c_i\leqq 0$ and $c_j\leqq 0$, then $c\leqq 0$ and $m_i\geqq m_j\geqq c_i+m_j$. Therefore, $m=m_i$ and $m_i\geqq m\geqq m_j$, i. e., $i\geqq J_\sigma\geqq j$. Suppose that $c_i\geqq 0$ and $c_j\leqq 0$. Then $m\geqq m_j$ and $c_i+m_j\geqq m_j$. Thus, if $c\leqq 0$, $i\geqq J_\sigma\geqq j$. If $c\geqq 0$ then it is evident that $J_\sigma\geqq j$. On the other hand, $c\leqq c_i$ holds. This implies $m_i-c_i\leqq m-c$ together with $m_i\leqq m$. Therefore, $i\geqq J_\sigma$. Suppose that $c_i\geqq 0$ and $c_j\geqq 0$. $c\geqq c_i\geqq 0$ must hold, and $m-c=\max\{m_i-c,\ m_j-c_j\}$. Notice that $m_i-c_i\geqq m_i-c$, and $m_i-c_i\leqq m_j-c_j=m-c$. Therefore, $i\geqq J_\sigma\geqq j$.

4. 4 A single machine scheduling problem minimizing the maximum lateness.

This problem has given a solution with computational requirement of O (n^2) under arbitrary precedence constraints (Lawler [5], see also [7]). It is shown below that a series-parallel algorithm with computational requirement of O ($n\log n$) can be applicable for this problem under series-parallel con-

straints.

S is the set of two-dimensional positive vectors (p, d), where p is the processing time and d is the due date of a job. (p_i, d_i) is element of S corresponding to an element i of N. The problem is to find a schedule which minimizes

$$f(\sigma) = \max \{\sum_{j=1}^{i} p_{\langle j\rangle} - d_{\langle i\rangle}\} .$$

An optimal schedule to this problem is given by arranging jobs according to a dominance relation defined as

$$i \leqq j \leftrightarrow d_i \leqq d_j ,$$

(Smith [14]).

This problem is a special case of the problem in 4. 3, where $c_i = p_i$, $m_i = p_i - d_i$ and $c_0 = 0$.

4. 5 A single machine scheduling problem minimizing total discounted cost.

In this section, $a[x]$ represents an exponential function with the base a and the exponent x. The problem P in this section is that in section 4. 1 with the following objective function.

$$f(\sigma) = -\sum_{i=1}^{n} \left\{ w_{\langle i\rangle} a\left[\sum_{j=1}^{i} p_{\langle j\rangle}\right]\right\} \qquad (0 \leqq a < 1) .$$

Because a is a nonnegative number less than 1, $f(\sigma)$ can be understood as the total of a kind of discounted costs. Let $C_{\langle i\rangle} = \sum_{j=1}^{i} p_{\langle j\rangle}$. The contribution of the i-th job to $f(\sigma)$, i. e., $-w_{\langle i\rangle} a[C_{\langle i\rangle}]$, rapidly increases as $C_{\langle i\rangle}$ does.

Define a dominance relation as

$$i \leqq j \leftrightarrow -\left(w_i a[p_i]\right)\Big/\left(1 - a[p_i]\right) \leqq -\left(w_j a[p_j]\right)\Big/\left(1 - a[p_j]\right) ,$$

and an optimal schedule will be given by arranging jobs according to the dominance relation (theorem 2 in Glazebrook and Gittens [2]). This dominance relation is recognized as a natural generalization of Smith's one that have been introduced in 4. 1 because

$$\lim_{a \to 1} (1-a) \cdot \frac{w_i a[p_i]}{1 - a[p_i]} = \frac{w_i}{p_i} .$$

Assumptions 1 and 2 are satisfied because Glazebrook-Gitten's dominace relation is evidently a total order.

Define σ_1, σ_2, σ_3 and σ'_2 as in 4. 1, then

$$f(\sigma_1 \cdot \sigma_2 \cdot \sigma_3) = -\sum_{i=1}^{k} w_i a[C_i] - \sum_{i=k+1}^{l-1} \left\{ w_i a[C_k + \sum_{j=k+1}^{i} p_j]\right\}$$
$$- \sum_{i=l}^{n} \left\{ w_i a[C_{l-1} + \sum_{j=l}^{i} p_j]\right\} .$$

Use here the identity

$$C_{l-1}=\sum_{j=1}^{l-1}p_j=C_k+\sum_{j=k+1}^{l-1}p_j=C_{\langle l-1\rangle},$$

then we will get

$$f(\sigma_1\cdot\sigma_2\cdot\sigma_3)=-\sum_{i-1}^{k}\left\{w_i a[C_i]\right\}-a[C_k]\sum_{i=k+1}^{l-1}\left\{w_i a[\sum_{j=k+1}^{i}p_j]\right\}$$
$$-\sum_{i=l}^{n}\left\{w_i a[C_{\langle l-1\rangle}+\sum_{j=l}^{i}p_j]\right\}.$$

Suppose that

$$f(\sigma_2)=-\sum_{i=k+1}^{l-1}\left\{w_i a[\sum_{j=k+1}^{i}p_j]\right\}$$
$$\leqq-\sum_{i=k+1}^{l-1}\left\{w_{\langle i\rangle}a[\sum_{j=k+1}^{i}p_{\langle j\rangle}]\right\}=f(\sigma'_2),$$

then, it is easy to obtain $f(\sigma_1\cdot\sigma_2\cdot\sigma_3)\leqq f(\sigma_1\cdot\sigma'_2\cdot\sigma_3)$, that is assumption 3 is satisfied. Define a composite job $J_{\sigma 2}=(p, w)$ corresponding to an arbitrary subschedule σ_2 as

$$p=\sum_{j=k+1}^{l-1}p_j,$$
$$w=\left(\sum_{i=k+1}^{l-1}\left\{w_i a[\sum_{j=k+1}^{i}p_j]\right\}\right)\Big/a[p].$$

Then, assumption 4 is satisfied because

$$f(\sigma_1\cdot\sigma_2\cdot\sigma_3)=-\sum_{i=1}^{k}\left\{w_i a[C_i]\right\}-a[C_k]\,wa[p]$$
$$-\sum_{i=1}^{n}\left\{w_i a[C_k+p+\sum_{j=1}^{i}p_j]\right\}=f(\sigma_1\cdot J_{\sigma 2}\cdot\sigma_3),$$

that is, $\Delta_{\sigma 2}=0$. Suppose next that $\sigma=i\cdot j$ and $i\geqq j$, then for J_σ

$$p=p_i+p_j,$$
$$w=\left(w_i a[p_i]+w_j a[p_i+p_j]\right)\Big/a[p_i+p_j].$$

It is simple algebraic calculations to show

$$-\left(w_i a[p_i]\right)\Big/\left(1-a[p_i]\right)\geqq-\left(wa[p]\right)\Big/\left(1-a[p]\right)$$
$$\geqq-\left(w_j a[p_j]\right)\Big/\left(1-a[p_j]\right).$$

Assumption 5 is also satisfied.

The problem in this section is treated in [2] and [13], where the results of Sidney [11] on the problem in 4. 1 were generalized. But, applicability of a series-parallel precedence constraints have not been clarified. Even a polynomial algorithm seems not to have been proposed.

5. SUMMARY

A sufficient condition was derived for a series-parallel algorithm to be applicable for permutation scheduling problems under series-parallel preced-

ence constaints. The power of the sufficient condition was demonstrated through five example problems.

Among them, those in 4. 1 and 4. 2 had already been shown to have series-parallel algorithms through inferences different from each other. Our analysis on them was rather simple and systematic. The problem in 4. 3 had already been given an algorithm with computational requirement less than or equal to O (n^2), and that in 4. 4 had already been given an algorithm of O (n^2) under general preceding constraints. But, applicability of series-parallel algorithms for them under series-parallel precedence constraints does not seem to have been shown. The problem in 4. 5 does not seem to have been given even a polynomial time algorithm under series-parallel precedence constraints.

It will be noteworthy that the problem in 4. 1 is NP-complete (Lawler [6]) and those in 4. 2 and 4. 3 are NP-hard (Monma [8]) under general precedence constraints. The problem in 4. 5 seems as at least difficult as the one in 4. 1.

Notice that we discussed a method to treat precedence constraints through theoretical development which is independent of specific problem properties such as attributes of jobs and objective functions. Most scheduling problems are NP-complete under general precedence constraints, even if they are in class P when no precedence constraint is imposed. The method in this paper suggests that it may be possible to clarify conditions, under which problems in class P does not become NP when general precedence constraints are imposed, from general properties of solution algorithms for cases without precedence constraints or of objective functions.

Sidney [13] studied problems under general precedence constraints in the form that is not dependent on specific properties of individual problems, though his interests are different from ours. It is worth studying relations of composite jobs to the interval order which is a basic tool of his theory.

REFERENCES

[1] Abdel-Wahab, H. M. and Kameda, T., "Scheduling to minimize Maximum Cumulative Cost Subject to Series-Parallel Precedence Constraints," Operations Research, **26**, 1 (1978), pp. 141–158.

[2] Glazebrook, K. D. and Gittins, J. C., "On Single-Machine Scheduling with Precedence Relations and Linear or Discounted Costs," Operations Research, **29**, 1 (1981), pp. 161–173.

[3] Johnson, S. M., "Optimal Two- and Three-Stage Production Schedules with Setup Times Included," Nav. Res. Logist. Quart., **1**, 1 (1954) pp. 61–68.

[4] Kurisu, T., "Two-Machine Scheduling under Required Precedence among Jobs," J. Opns. Res. Soc. Japan, **19**, 1 (1976), pp. 1–13.

[5] Lawler, E. L., "Optimal Sequencing of a Single Machine Subject to Precedence

Constraints," Management Science, **19**, 5 (1973), pp. 544–546.

[6] Lawler, E. L., "Sequencing Jobs to Minimize Total Weighted Completion Time Subject to Precedence Constraints," Annals of Discrete Mathematics, **2**, (1978), pp. 75–90.

[7] Graham, R. L., Lawler, E. L., Lenstra, J. K. and Rinnooy Kan. A. H. G., "Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey," MC Tract 99, Mathematisch Centrum, Amsterdam (1978), pp. 169–231.

[8] Monma, C. L., "The Two-Machine Maximum Flow Time Problem with Series-Parallel Precendence Constraints: an Algorithm and Extensions," Operations Research, **27**, 4 (1979), pp. 792–798.

[9] Sekiguchi, Y., "Inter- and Intra-Group-of-Jobs Schedule for Minimizing Makespan in a Two-Machine GT Shop," Preprints of IFAC 8th Trienial World Congress, Kyoto, Japan. (1981), pp. XIV–164–XIV–169.

[10] Sekiguchi, Y., "Optimal Schedules under Precedence Constraints in a GT-Type Flow-Shop," Manuscript (March, 1982).

[11] Sidney, J. B., "Decomposition Algorithms for Single-Machine Sequencing with Precedence Relations and Deferral Costs," Operations Research, **23**, 2 (1975), pp. 283–298.

[12] Sidney, J. B., "The Two-Machine Maximum Flow Time Problem with Series-Parallel Precedence Relations," Operations Research, **27**, 4 (1979), pp. 782–791.

[13] Sidney, J. B., "A Decomposition Algorithm for Sequencing with General Precedence Constraints," Mathematics of Operations Research, **6**, 2 (1981). pp. 190–204.

[14] Smith, W. E., "Various Optimizers for Single-Stage Production," Naval. Res. Logist. Quart., **3**, 1 (1956), pp. 59–66.

(1982. 4. 13)