Single Machine Scheduling under Series-Parallel Precedence Constraints

Valery Gordon
Institute of Engineering Cybernetics, National Academy of Sciences of Belarus
gordon@newman.bas-net.by

Vitaly Strusevich
University of Greenwich, London, UK
V.Strusevich@greenwich.ac.uk

Abstract
We consider the single machine due date assignment and scheduling problems under series-parallel precedence constraints and some properties of series-parallel graphs.

1. Introduction

The problems under consideration arise in production planning when the management is faced with a problem of setting realistic due dates for a number of jobs. The due dates are determined by increasing the time needed for the jobs fulfillment by a common positive slack. The objective is to explore the trade-off between the size of the slack and the arising holding costs for the early jobs. A schedule is feasible if there are no tardy jobs and the job sequence respects given precedence constraints. Imposing the precedence constraints over set of jobs to describe possible sequences of jobs is often used to deal with situations when products are manufactured in a certain order implied, for example, by technological, marketing or assembly requirements. Among the factors that may affect the complexity of scheduling problem are the structure of precedence constraints and the objective function involved. We consider the case when precedence constraints are defined by series-parallel graph and propose polynomial time algorithms for some scheduling problems with due date assignment.

2. Properties of series-parallel graphs

Formally, the precedence constraints among the jobs are defined by a binary relation $\to$. We write $J_i \to J_j$ and say that job J_i *precedes* J_j if in any feasible schedule job J_i must be completed before J_j starts. We write $J_i \sim J_j$ if jobs J_i and J_j are independent. The precedence constraints are usually given by a directed circuit-free graph *G* in which the set of vertices is identical with the set *N* of jobs and there is the arc from vertex J_i to vertex J_j if and only if $J_i \to J_j$. The graph obtained from *G* by removing all transitive arcs is called the *reduction graph* and is denoted by $\vec{G}$. A sequence (or a permutation) of jobs is *feasible* if no pair of jobs violate the precedence constraints.

Let $G(X,U)$ be a (di)graph, where *X* is the set of vertices and *U* is the set of arcs. A graph $G(X,U)$ is said to be *a parallel composition* of two graphs $G_1(X_1,U_1)$ and $G_2(X_2,U_2)$ such that $X_1 \cap X_2 = 0$, if $X = X_1 \cup X_2$ and $U = U_1 \cup U_2$. A graph $G(X,U)$ is said to be *a series composition* of two graphs $G_1(X_1,U_1)$ and $G_2(X_2,U_2)$ such that $X_1 \cap X_2 = 0$, if $X = X_1 \cup X_2$ and $U = U_1 \cup U_2 \cup \tilde{U}$, where $\tilde{U}$ is the set of arcs from each vertex of the graph G_1 with zero outdegree to each vertex of the graph G_2 with zero indegree. A graph is called *series-parallel* (or SP-*graph*) if either it consists of only one vertex, or it can be obtained from a set of single-vertex graphs by a subsequent application of the operations of series and/or parallel composition. Sometimes, the graph obtained from an SP-graph by removing all its transitive arcs is also called series-parallel. In what follows, we consider the graphs with no transitive arcs.

Let k be a vertex of graph $G(X,U)$ and $A(k)$ be a set of all vertices $i \in X$ such that $k \to i$. Let $B(k)$ be a set of all vertices $i \in X$ such that $i \to k$. We say that graph $G(X,U)$ has property **A** if $A(i) \setminus B(k) = A(j) \setminus B(k)$ for any vertex $k \in X$ and any $i, j \in B(k)$ such that $i \sim j$. We say that graph $G(X,U)$ has property **B** if $B(i) \setminus A(k) = B(j) \setminus A(k)$ for any vertex $k \in X$ and any $i, j \in A(k)$ such that $i \sim j$.

Then the following theorem is valid [1].

Theorem. The following statements are equivalent for a graph $G(X,U)$ with no transitive arcs:

(a) Graph G is a series-parallel;

(b) Graph G has property **A**;

(c) Graph G has property **B**;

(d) Transitive closure of graph G does not contain a subgraph which is isomorphic to Z-graph (see Figure 1).

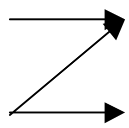

Figure 1. Z-graph

Based on these properties, the recognition of series-parallel graphs can be done in $O(|X|^2)$ time [6, 8]. An SP-graph is sometimes given by its so-called *decomposition tree* which can be found in $O(|X|^2)$ time. See [4, 7, 8] for the definition of the decomposition tree and for further details on the SP-graphs.

3. Scheduling under series-parallel precedence constraints

The structural properties of series-parallel graphs allow the designing of fast algorithms for minimizing a wide range of functions over partially ordered sets.

Let $p^{ab} = (p^1, p^a, p^b, p^2)$ and $p^{ba} = (p^1, p^b, p^a, p^2)$ be two permutations of n elements that differ only in the order of the subsequences p^a and p^b. For a function $F(p)$ that depends on a permutation, suppose that there exists a function $w(p)$ such that for any two permutations $p^{ab} = (p^1, p^a, p^b, p^2)$ and $p^{ba} = (p^1, p^b, p^a, p^2)$ the inequality

$w(p^a) > w(p^b)$ implies that $F(p^{ab}) \leq F(p^{ba})$, while the equality $w(p^a) = w(p^b)$ implies that $F(p^{ab}) = F(p^{ba})$. In this case, function *F* is called *a priority-generating function*, while function w is called its *priority function*. For a (partial) permutation p, the value of $w(p)$ is called *the priority* of that permutation.

As described in [5, 7], any priority-generating function can be minimized in $O(n \log n)$ time if the reduction graph $\vec{G}$ is series-parallel and is given by its decomposition tree. One of the examples of priority-generating functions arises in the single machine scheduling problem of minimizing the total weighted completion time $\sum w_j C_j$, where C_j denotes the completion time of job J_j, and w_j is the job's weight. If $p = (p(1), p(2), ..., p(n))$ is a sequence of job indices with index $p(k)$ that appears in the *k*th position, then the objective function can be written as $F(p) = \sum_{k=1}^{n} w_{p(k)} \sum_{j=1}^{k} p_{p(j)}$, where p_i denotes the processing time of job J_i. This function is priority-generating and for a partial permutation s the priority function is $w(s) = w(s) / p(s)$, where $w(s)$ and $p(s)$ denote the sum of the weights and the processing times, respectively, of all jobs whose indices are involved in s.

We consider the following single machine scheduling problem. The jobs of the set $N = \{J_1, ..., J_n\}$ have to be processed with no preemption on a single machine, and the processing of job J_j takes p_j time units, $1 \leq j \leq n$. The jobs are simultaneously available at time zero, and for each job J_j a weight w_j is given that indicates its relative importance. The machine can handle only one job at a time, and is permanently available from time zero. The machine is allowed to be idle if required. Each job J_j has the *due date* d_j by which it is desirable to complete its processing. Job J_j is *tardy* if $C_j > d_j$, and job is *early* if $C_j < d_j$ and $E_j = d_j - C_j$ is its *earliness*. We assume that the due dates are determined according to the so-called SLK rule [2], i.e., for each job the due date is obtained by adding a positive slack *q* to the processing time: $d_j = p_j + q$, $j = 1, ..., n$. We look for the value of *q* which minimizes the function $j(F, q)$ over the set of schedules in which no job is tardy. Here $j(F, q)$ is an arbitrary non-decreasing function in both arguments, and *F* is either $\sum w_j E_j$ or $\sum w_j \exp(gE_j)$, where $g \neq 0$. Once *q* is chosen or fixed, the corresponding scheduling problem is to find a feasible schedule with the minimum value of function *F*. The functions $F = \sum w_j E_j$ and $F = \sum w_j \exp(gE_j)$ are closely related to priority-generating functions. Using general scheme [3] for the due date assignment problems with arbitrary precedence constraints and arbitrary functions *F*, we propose $O(n^2 \log n)$ algorithms for series-parallel precedence constraints provided that *F* is either the sum of linear functions or the sum of exponential functions. The running time of algorithms can be reduced to $O(n \log n)$ if the jobs are independent.

Acknowledgements

This work for the first author was partly supported by INTAS (Project 00-217) and by ISTC.

References

[1] Gordon, V.S. (1981). Some properties of series-parallel graphs. *Izvestiya Akademii Nauk BSSR (Proceedings of the Academy of Sciences of BSSR)*, 1, 18-23 (in Russian).

[2] Gordon, V.S., Proth, J.-M. and Chu, C. (2001). A state-of-the-art survey of due date assignment and scheduling research: SLK, TWK, and other due date assignment models, (to appear in *Production Planning and Control*). See also INRIA Rapport de Recherche 3537, 1998, 1-37.

[3] Gordon, V.S. and Strusevich, V.A. (1999). Earliness penalties on a single machine subject to precedence constraints: SLK due date assignment. *Computers & Operations Research*, 26, 157-177.

[4] Lawler, E.L. (1978). Sequencing jobs to minimize total weighted completion time subject to precedence constraints. *Annals of Discrete Mathematics*, 2, 75-90.

[5] Monma, C.L. and Sidney, J.B. (1979). Sequencing with series-parallel precedence constraints. *Mathematics of Operations Research*, 4, 215-234.

[6] Shafransky, Y.M. and Yanova, O.V. (1980). The recognition and decomposition of series-parallel graphs. In: *Library of the Programs for Solving Extremal Problems*. Minsk, IEC of the AS BSSR, 17-24 (in Russian).

[7] Tanaev, V.S., Gordon, V.S. and Shafransky, Y.M. (1994). *Scheduling Theory. Single-Stage Systems*. Kluwer Academic, Dordrecht et al.

[8] Valdes, J.R., Tarjan, E. and Lawler E.L. (1982). The recognition of series-parallel digraphs. *SIAM Journal on Computing*, 11, 361-370.