SCHEDULING

Peter Brucker and **Sigrid Knust**
University of Osnabrück, Germany

Keywords: scheduling, project scheduling, machine scheduling, timetabling, classification scheme, complexity, polynomially solvable, NP-hard, dynamic programming, branch-and-bound methods, local search methods, genetic algorithms, lower bounds, approximation algorithms.

Contents

1. Introduction
2. General Scheduling Models
3. Applications
3.1. Project Scheduling
3.2. Machine Scheduling Problems
3.3. Timetabling
4. Classification, Complexity and Solution Methods
4.1. A Classification Scheme
4.2. Polynomially Solvable Scheduling Problems
4.3. NP-hard Scheduling Problems
4.4. Complexity of Single-Machine Problems
4.5. Living with NP-hard Problems
Glossary
Bibliography
Biographical Sketches

Summary

This chapter presents a survey on scheduling models and some of their applications (project scheduling, machine scheduling and timetabling). All these models have in common that certain activities requiring some scarce resources have to be planned over the time. Depending on the complexity of the problems, different solution methods are presented.

1. Introduction

Scheduling is concerned with the allocation of limited resources to activities over time. The activities may be tasks in a construction project, operations in a production process, lectures at the university, and so on. The resources may be workers, machines, lecturers, and so on. General scheduling models will be introduced and specific applications like project scheduling, machine scheduling, and timetabling will be discussed.

Methods for solving scheduling problems depend on the computational complexity. For project and machine scheduling problems, a sophisticated classification scheme has been introduced. Such a scheme for scheduling problems is briefly described and polynomial solvable and $\mathcal{NP}$-hard problems are introduced. Finally, some methods to

cope with $\mathcal{NP}$-hard scheduling problems are described.

2. General Scheduling Models

A general scheduling problem is the resource-constrained project scheduling problem (RCPSP) which can be formulated as follows:

Given are n activities (tasks, jobs, operations, lectures) $j = 1, \ldots, n$ and r (renewable) resources $k = 1, \ldots, r$. R_k units of resource k are available in each time period $[t, t + 1[$ for $t = 0, \ldots, T - 1$, where T denotes a given time horizon. Activity j must be processed for p_j time units without interruption. Thus, if S_j denotes the starting time of activity j, it completes at time $C_j = S_j + p_j$. During this time period $[S_j, C_j[$ a constant amount of r_{jk} units of resource k ($k = 1, \ldots, r$) is occupied. Furthermore, precedence constraints $i \rightarrow j$ are defined between certain activities i, j. The meaning of $i \rightarrow j$ is that activity j cannot start before activity i is completed, i.e. $S_i + p_i \leq S_j$ must hold. A schedule $S = (S_j)$ is defined by the starting times S_j of all activities. S is called feasible if

- in each time period $[t, t + 1[$ the total resource demand for each resource k is less than or equal to the availability R_k, and,
- the given precedence constraints are satisfied.

One has to find a feasible schedule S such that the makespan

$$C_{\max} := \max_{j=1}^{n}\{C_j\} = \max_{j=1}^{n}\{S_j + p_j\}$$

is minimized.

There are several possible ways to add constraints or to generalize the RCPSP.

Time-dependent resource profiles

Instead of constant availabilities R_k the availability of resource k may be given by a function $R_k(t)$ depending on the periods $[t, t + 1[$. This includes the nonavailability of resource k if $R_k(t) = 0$ for certain t-values. One has to find a feasible schedule within the time horizon $[0, T]$. If such a schedule exists, one has to find a feasible schedule which minimizes the makespan. A resource k with $R_k(t) \in \{0, 1\}$ is called disjunctive, otherwise it is called cumulative.

Example 1

Consider an instance of a project with $n = 6$ *activities and* $r = 2$ *resources, precedence constraints* $1 \rightarrow 4 \rightarrow 5, 2 \rightarrow 3$ *and the following data*

j	1	2	3	4	5	6
p_j	2	2	3	2	2	4

r_{j1}	1	0	1	0	1	0
r_{j2}	1	2	3	2	4	2

Figure 1 shows the time-dependent resource profiles and represents a feasible schedule by the corresponding Gantt chart. This schedule does not minimize the makespan because by moving activity 3 two units to the right and scheduling activity 6 between activity 1 and activity 3 a feasible schedule with smaller makespan is obtained.

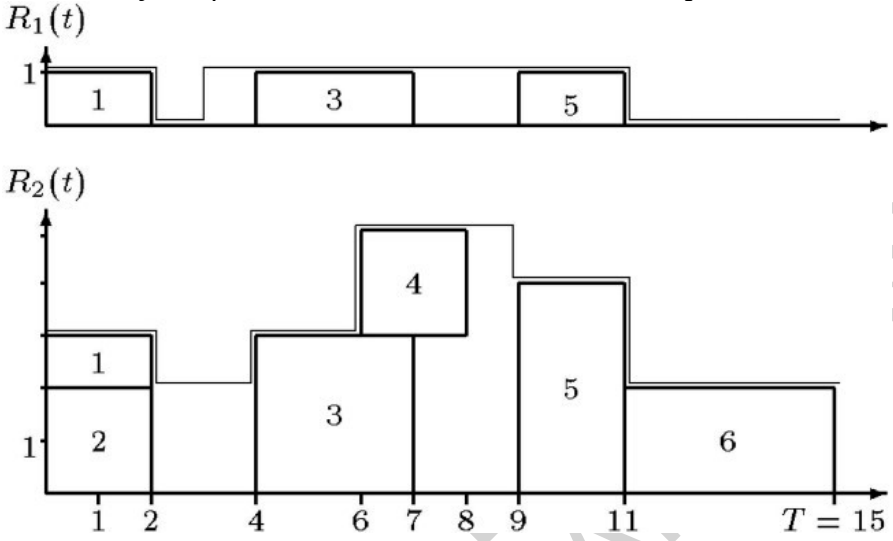


Figure 1: A feasible schedule

Preemptions

The assumption that each activity j must be processed within an interval $[S_j, S_j + p_j[$ may be relaxed by allowing preemptions. Preemption of an activity means that processing may be interrupted and resumed at a later time. In this case, preemptions and continuations of activities are allowed at integer times only.

Parallelity constraints

Two activities may be forced to be processed in parallel for at least one time unit.

Minimal and maximal time-lags

A precedence relation $i \rightarrow j$ may be replaced by a start-start relation of the form $S_i + d_{ij} \leq S_j$ where d_{ij} is an arbitrary integer number. The interpretation of this relation depends on the sign of d_{ij}. If $d_{ij} \geq 0$, then activity j cannot start before d_{ij} time units after the start of activity i. This means that activity j does not start before the starting time of activity i and d_{ij} is a minimal distance between both starting times (Figure 2(a)). If on the other hand, $d_{ij} < 0$, then the earliest start of activity j is $-d_{ij}$ time units before the start of activity i, i.e. activity i cannot start more than $-d_{ij}$ time units later than the starting time of activity j. If $S_j \leq S_i$ this means that $-d_{ij}$ is a maximal distance between both starting times (Figure 2(b)).

Such relations are very general timing relations between activities. For example, with $d_{ij} = p_i$ it is equivalent to the precedence relation $i \rightarrow j$. More generally, if there should be a minimal time distance of l_{ij} units between the completion of activity i and the start of activity j, then $S_i + p_i + l_{ij} \leq S_j$. If $S_i + p_i + l_{ij} \leq S_j$ and $S_j - u_{ij} - p_i \leq S_i$ hold where $0 \leq l_{ij} \leq u_{ij}$, then the time between the completion time of activity i and the starting time of activity j must be at least l_{ij} but no more than u_{ij}. This includes the special case $0 \leq l_{ij} = u_{ij}$ where activity j must start exactly l_{ij} time units after the completion of activity i.

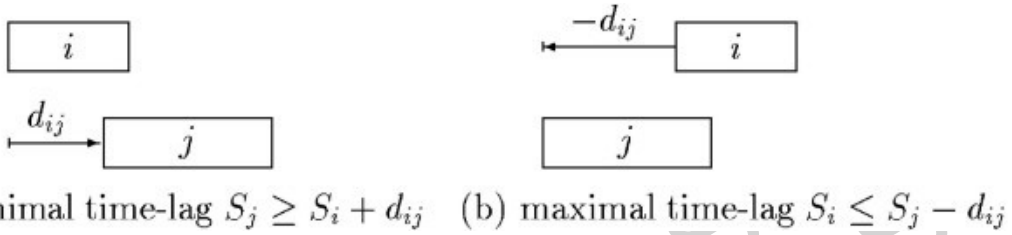


Figure 2: Minimal and maximal time-lags.

Separating times between activities

Given two activities i and j, the condition $S_i + d_{ij} \leq S_j$ may be replaced by the weaker condition that $S_i + d_{ij} \leq S_j$ or $S_j + d_{ji} \leq S_i$ has to be satisfied. If $d_{ij} = p_i + l$ and $d_{ji} = p_j + l$ with $l \geq 0$, then between the completion time of one of the two activities i, j and the starting time of the other activity a minimal separating time of l time units has to be respected.

Bounding restrictions

Let B be a set of activities and U be a set of time periods. Then it may be required that the activities in B are processed for at least l and for at most u time units within the time periods defined by U, where l and u are non-negative integers with $l \leq u$. The settings $l = 0$ or $u = \infty$ impose no restriction.

Multi-modes

In the multi-mode case, a set M_j of modes (processing alternatives) is associated with each activity j. The processing time of activity j in mode m is given by p_{jm} and per period usage of renewable resource k is given by r_{jkm}. Furthermore, so-called non-renewable resources may be given in the multi-mode case. While for the renewable resources the capacity is limited for each time period (like machines, people), the non-renewable resources are capacitated over the whole time horizon (like money, energy) and are consumed by the activities. One has to assign a mode to each activity and to schedule the activities in the assigned mode.

Other objective functions

Besides the objective of minimizing the makespan $C_{\max} := \max_{j=1}^{n}\{C_j\}$ one may consider other objective functions $f(C_1, \ldots, C_n)$ depending on the completion times of the activities. Examples are the total flow time $\sum_{j=1}^{n} C_j$ or more generally the weighted (total) flow time $\sum_{j=1}^{n} w_j C_j$. Other objective functions depend on due dates d_j which are associated with the activities. With the lateness $L_j := C_j - d_j$, the tardiness $T_j := \max\{0, C_j - d_j\}$ and the unit penalty $U_j := \begin{cases} 0 \text{ if } C_j \le d_j \\ 1 \text{ otherwise} \end{cases}$ the objective functions

$L_{\max} := \max_{j=1}^{n}\{L_j\}$ (maximum lateness),

$\sum_{j=1}^{n} T_j$ (total tardiness),

$\sum_{j=1}^{n} w_j T_j$ (total weighted tardiness),

$\sum_{j=1}^{n} U_j$ (number of late activities),

$\sum_{j=1}^{n} w_j U_j$ (weighted number of late activities),

may be defined. All these objective functions are regular, i.e. monotone non-decreasing functions of the completion times C_j. Other even nonregular objective functions are possible. In connection with project scheduling problems also objectives concerning resource leveling or resource investment are common. In this case, the resource capacities are not given, but have to be determined (inducing some additional costs).

3. Applications

The basic models introduced in the previous section cover a wide range of applications. These applications are characterized on one side by special characteristics of the basic model. On the other side, special features and additional restrictions may be added to the basic model. In this section three of the most important applications, namely project scheduling, machine scheduling, and timetabling are discussed.

3.1. Project Scheduling

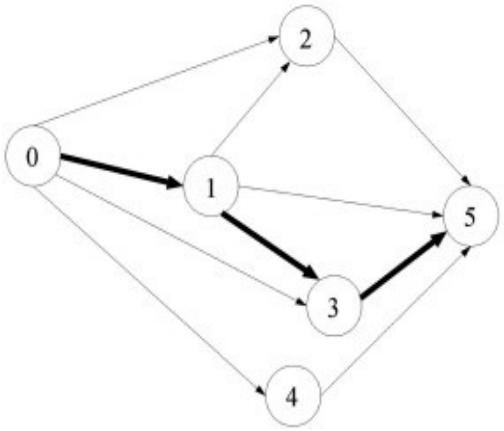
Scheduling of projects has many faces, e.g. the design of production facilities, construction or maintenance projects, new product development and introduction, installation of computer hardware or software, etc. Different models and solution methods to solve the corresponding scheduling problems have been developed.

As previously described, a project consists of a set of activities, a set of resources and precedence relations $i \to j$ between some activities. It is convenient to introduce a dummy starting-activity 0 and a dummy end-activity $n + 1$ requiring no resources with processing times $p_0 = p_{n+1} = 0$. Furthermore, $0 \to j \to n+1$ holds for all regular activities $j = 1, \ldots, n$. Associated with a project is an acyclic directed graph $G = (V, A)$ in which the nodes $j \in V$ are the activities and the arc set A represents the set of all precedence relations $i \to j$. The project is defined by this graph G, the resource availabilities R_k, and the data p_j, r_{jk} of all activities j.

If $R_k = \infty$ for all resources k, i.e. if there are no resource restrictions, the earliest starting time est_j of each activity j can be calculated efficiently by computing the length l_j of a longest path from the starting activity 0 to activity j in the directed graph $G = (V, A)$. The length of a (directed) path P in G is the sum of all processing times of activities in P, the last one excluded. It is easy to show that $S = (S_j)$ with $S_j = l_j$ for all activities j is a feasible schedule. A longest path from 0 to $n + 1$ is called a critical path. The length l_{n+1} of a critical path is equal to the minimal makespan if there are no resource restrictions. In the general RCPSP (with resource restrictions), the values l_j are only lower bounds for the starting times S_j of the activities j. Symmetrically, the length q_j of a longest path from activity j to the end-activity $n + 1$ can be calculated. Then, $T - q_j$ is an upper bound for the latest starting time lst_j of j in any feasible schedule with makespan $C_{max} \leq T$. To find a schedule for the RCPSP which minimizes the makespan is a more difficult problem.

Example 2

In Figure 3, a small project with n = 4 activities and sufficient resources (i.e. $R_k = \infty$ for all resources k) is defined. A corresponding critical path is $0 \to 1 \to 3 \to 5$ with length $l_5 = 14$. Furthermore, the earliest and latest starting times est_j and lst_j are listed for $T = l_5 = 14$.



j	0	1	2	3	4	5
p_j	0	5	6	9	4	0
r_{j1}	0	1	2	2	2	0
r_{j2}	0	5	2	4	3	0
est_j	0	0	5	5	0	14
lst_j	0	0	8	5	10	14

Figure 3: Earliest and latest starting times for a project with $n = 4$

-
-
-

Bibliography

Aarts, E. and Lenstra J.K. (Ed.). (1997). *Local Search in Combinatorial Optimization*, 512 pp. Chichester, New York: Wiley. [Collection of various articles about local search (theoretical and practical)]

Błażewicz, J., Ecker, K., Pesch, E., Schmidt, G. and Węglarz J. (1996). *Scheduling Computer and Manufacturing Processes*, 491 pp., Berlin: Springer. [Text book on scheduling, contains several applications from computer and manufacturing environments]

Brucker, P. (1998). *Scheduling Algorithms*, 2nd Edition, 342 pp., Berlin: Springer. [Text book on machine scheduling, covers especially polynomial algorithms]

Brucker, P., Drexl, A., Möhring, R., Neumann, K. and Pesch, E. (1999). Resource-Constrained Project Scheduling: Notation, Classification, Models and Methods. *European Journal of Operational Research* **112**, 3-14. [A survey on recent developments in project scheduling]

Brucker, P. and Knust, S. (1998). *Complexity results for scheduling problems*. <http://www.mathematik.uni-osnabrueck.de/research/OR/class>. [Collection of complexity tables for several classes of scheduling problems]

Chen, B., Potts, C.N., and Woeginger, G.J. (1998). A Review of Machine Scheduling: Complexity, Algorithms and Approximability. *Handbook on Combinatorial Optimization* (eds. Ding-Zhu Du, and P.M. Pardalos). Boston: Kluwer Academic Publishing. [Extensive survey of complexity and approximation results for machine scheduling problems]

Garey, M. R. and Johnson, D.S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. 338pp. San Francisco: W.H. Freeman and Company. [The main introduction to complexity theory and NP-hardness]

Hoogeveen, J.A., Lenstra, J.K., and van de Velde, S.L. (1997). *Sequencing and Scheduling, in: Annotated Bibliographies in Combinatorial Optimization* (eds. M. Dell' Amico, T. Maffioli, and S. Martello). Wiley. [Annotated bibliography on scheduling]

Pinedo, M. (1995). *Scheduling: Theory, Algorithms, and Systems*, 378 pp. Englewood Cliffs, New Jersey: Prentice Hall. [Text book on machine scheduling, covers also stochastic models and practical applications, contains several exercises]

Węglarz, J. (ed.) (1999). *Project Scheduling: Recent Models, Algorithms and Applications*, 535 pp. Boston: Kluwer. [Collection of various articles covering the current research in project scheduling]

Biographical Sketches

Peter Brucker is Professor for Applied Mathematics at the Department of Mathematics and Computer Science of the University of Osnabrück, Germany. He received his diploma and doctoral degrees at the Free University of Berlin. His main research areas are discrete optimization and scheduling. He is the author of about 100 scientific publications, among them four monographs. Moreover, he is an Associate Editor of five scientific journals, and was organizer of several international conferences.

Sigrid Knust is Junior Professor at the Department of Mathematics and Computer Science of the University of Osnabrück, Germany. She received her diploma and doctoral degree at the University of Osnabrück, Department of Mathematics and Computer Science. Her research area is discrete optimization, especially scheduling.