ELSEVIER

European Journal of Operational Research 80 (1995) 94–102

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

Theory and Methodology

# Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem

Arno Sprecher, Rainer Kolisch *, Andreas Drexl

*Institut für Betriebswirtschaftslehre, Christian-Albrechts-Universität zu Kiel, Ohlshausenstr. 40, 24118 Kiel, Germany*

Received March 1993; revised July 1993

**Abstract**

We consider the resource-constrained project scheduling problem (RCPSP). The focus of the paper is on a formal definition of semi-active, active, and non-delay schedules. Traditionally these schedules establish basic concepts within the job shop scheduling literature. There they are usually defined in a rather informal way which does not create any substantial problems. Using these concepts in the more general RCPSP without giving a formal definition may cause serious problems. After providing a formal definition of semi-active, active, and non-delay schedules for the RCPSP we outline some of these problems occurring within the disjunctive arc concept.

*Keywords:* Resource-constrained project scheduling; Semi-active, active, and non-delay schedules; Branch-and-bound methods

## 1. Introduction

Classification of schedules is the basic work to be done in order to attack scheduling problems. For the case of the job shop problem (JSP) thorough studies have been performed (cf. Conway et al., 1967; Baker, 1974; Rinnooy Kan, 1976; French, 1982). Schedules for the JSP are classified as feasible, semi-active, active, and non-delay schedules. Procedures minimizing a regular measure of performance are usually enumerating semi-active or active schedules (cf. Rinnooy Kan, 1976). The latter are known to be the smallest dominant set of schedules (cf. French, 1982; Rinnooy Kan, 1976).

For the resource-constrained project scheduling problem (RCPSP) as a generalization of the JSP the majority of researchers did not make use of any schedule classification (cf. e.g. Davis and Heidorn, 1971; Stinson et al., 1978; Talbot and Patterson, 1978; Talbot, 1982; Patterson et al., 1989). Some researchers just defined the type of schedule needed. Thereby different definitions have been proposed for the same type of schedules and identical definitions have been used for different kinds of schedules. E.g. Elmaghraby

* Corresponding author.


*SSDI* 0377-2217(93)E0294-8

(1977, p. 205) defines an *eligible schedule* as a schedule where no activity can be started earlier without changing the start times of any other activity and still maintain feasibility. In Schrage (1970) the same type of schedule is called an *active schedule*.

Wiest defines a *left-justified schedule* as a "feasible schedule in which...no job can be started at an earlier date by local left shifting of that job alone" (Wiest, 1964) whereas Gonguet calls a schedule left-justified if "each job is scheduled as early as possible" (Gonguet, 1969).

Finally other researchers have just taken over the schedule classification of the JSP without modifications (cf. Radermacher, 1985/1986; Demeulemeester and Herroelen, 1992a; Herroelen and Demeulemeester, 1992). This is somewhat reasoned by the way schedule classification is presented in the textbooks for the JSP most oftenly cited (cf. Baker, 1974; French, 1982). There the definitions are more illustrative than formal and thus bear ambiguity in the case of the more general RCPSP.

In order to present a general and precise schedule classification we proceed as follows: Using the schedule classification for the JSP proposed by Baker (1974) as the stepping stone, we develop more formal and general definitions, that is, schedules for the RCPSP are discriminated to be feasible, semi-active, active, or non-delay schedules. Naturally this classification holds for the JSP as well. Moreover, the (dominance) relations between the different sets of schedules, as known from the JSP, are preserved. This will help to classify procedures for the RCPSP based on the schedules they examine.

The remainder of the paper is organized as follows: In Section 2 we give a formal description of the RCPSP. The classification of schedules for the RCPSP is dealt with in Section 3. Section 4 then provides examples in order to illustrate the definitions. Furthermore, in Section 5 some problems which arise by the adaptation of the disjunctive arc concept (Balas, 1969) to the RCPSP (Radermacher, 1985/1986; Cristofides et al., 1987; Bell and Park, 1990) are demonstrated with examples. Finally, conclusions are drawn in Section 6.

## 2. Problem description

The classical resource-constrained project scheduling problem can be stated as follows: We consider a single project which consists of $J$ activities, $j = 1$ ($j = J$) being the unique dummy source (sink). The activities are partially ordered by precedence relations, where $P_j$ is the set of immediate predecessors of activity $j$. Furthermore, the activities are numerically labeled, i.e. a predecessor of $j$ has a smaller number than $j$. The precedence relations between the activities can be represented by an acyclic activity-on-node network (AON). The set of (renewable) resources is refered to with $R$. Each $r \in R$ has a limited period availability of $K_{rt}$ units in period $t$, $t = 1,\ldots,T$, where $T$ denotes an upper bound on the project's makespan. The activity $j$, $1 \leq j \leq J$, has a duration of $d_j$ periods ($d_1 = d_J = 0$) and uses $k_{jr}$ units of resource $r$, $r \in R$, each period it is in process. All parameters are supposed to be integer valued. Table 1 provides a summary of the symbols and definitions.

For modelling purposes we use binary variables $x_{jt}$, $j = 1,\ldots,J$, $t = \mathrm{EF}_j,\ldots,\mathrm{LF}_j$,

$$x_{jt} := \begin{cases} 1 & \text{if job } j \text{ is finished at time } t, \\ & \text{i.e. at the end of period } t, \\ 0, & \text{otherwise,} \end{cases}$$

as proposed in Pritsker et al., (1969). The earliest finish times, $\mathrm{EF}_j$, and latest finish times, $\mathrm{LF}_j$, are calculated by traditional forward recursion and backward recursion with $\mathrm{LF}_j := T$.

Table 1
Symbols and definitions

| | |
|---|---|
| $j = 1,\ldots,J$ | activities of a (single) project |
| $j = 1$ ($j = J$) | unique dummy source (unique dummy sink) |
| $d_j$ | duration of activity $j$ (in periods) |
| $P_j$ | set of predecessors of activity $j$ |
| $\mathrm{EF}_j$ ($\mathrm{LF}_j$) | earliest (latest) finish time of activity $j$ |
| $\mathrm{ST}_j$ ($\mathrm{FT}_j$) | start (finish) time of activity $j$ |
| $r \in R$ | renewable resource |
| $k_{jr}$ | job $j$ uses $k_{jr}$ units of resource $r$ each period it is in process |
| $K_{rt}$ | availability of resource $r$ in period $t$ |
| $T$ | upper bound on the project's makespan |
| $t = 1,\ldots,T$ | period |

Table 2
Constraints of the RCPSP

$$\sum_{t=\mathrm{EF}_j}^{\mathrm{LF}_j} x_{jt} = 1 \qquad j = 1,\ldots,J \qquad (1)$$

$$\sum_{t=\mathrm{EF}_i}^{\mathrm{LF}_i} t x_{it} \le \sum_{t=\mathrm{EF}_j}^{\mathrm{LF}_j} (t-d_j) x_{jt} \qquad j = 2,\ldots,J,\ i \in P_j \qquad (2)$$

$$\sum_{j=1}^{J} k_{jr} \sum_{\tau=t}^{t+d_j-1} x_{j\tau} \le K_{rt} \qquad r \in R,\ t = 1,\ldots,T \qquad (3)$$

$$x_{jt} \in \{0,1\} \qquad j = 1,\ldots,J,\ t = \mathrm{EF}_j,\ldots,\mathrm{LF}_j \qquad (4)$$

The constraints are given in Table 2. Equation (1) ensures that each activity is assigned exactly one finish time within the interval $[\mathrm{EF}_j, \mathrm{LF}_j]$. Equation (2) secures that precedence relations are maintained and (3) limits the period usage of resources to their availability. With $\mathrm{ST}_j$ ($\mathrm{FT}_j$) we denote the start (finish) *time* of activity $j$, where activity $j$ is scheduled in the *periods* $\mathrm{ST}_j + 1,\ldots,\mathrm{ST}_j + d_j = \mathrm{FT}_j$. In the example provided by Fig. 1 we have $\mathrm{ST}_2 = 0$ ($\mathrm{FT}_2 = 2$) and activity 2 is scheduled in periods one and two.

As objective functions we consider only regular measures of performance.

**Definition 1** (Conway et al., 1967; Rinnooy Kan, 1976). Let $\mathrm{FT}_1,\ldots,\mathrm{FT}_J$ be the finish times of activities $1,\ldots,J$, respectively. A performance measure is a mapping

$$\phi : Z^J_{\ge 0} \to R_{\ge 0}$$

which assigns to each $J$-tuple $(\mathrm{FT}_1,\ldots,\mathrm{FT}_J)$ of finish times a performance value $\phi(\mathrm{FT}_1,\ldots,\mathrm{FT}_J)$. If $\phi$ is monotonically increasing with respect to componentwise ordering, that is, $\phi(\mathrm{FT}_1,\ldots,\mathrm{FT}_J) > \phi(\mathrm{FT}'_1,\ldots,\mathrm{FT}'_J)$ implies

$$\mathrm{FT}_j \ge \mathrm{FT}'_j \quad \forall j,\ 1 \le j \le J,$$

$$\mathrm{FT}_j > \mathrm{FT}'_j \quad \exists j,\ 1 \le j \le J,$$

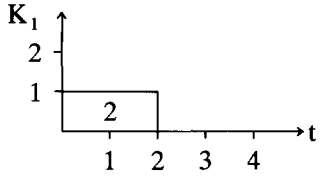


Fig. 1. Relationship between time and period.

and in addition minimization is considered, then we call the performance measure regular.

The most commonly considered regular performance measure for the RCPSP is the minimization of the makespan:

$$\text{minimize} \sum_{t=\mathrm{EF}_J}^{\mathrm{LF}_J} t x_{Jt}. \qquad (5)$$

Another one is the minimization of the mean flow time:

$$\text{minimize} \frac{1}{J-2} \sum_{j=2}^{J-1} \sum_{t=\mathrm{EF}_j}^{\mathrm{LF}_j} t x_{jt}. \qquad (6)$$

Further regular measures for the RCPSP can be found in, e.g. Slowinski (1989) and Patterson et al. (1990).

For ease of notation and the sake of readability we have presented the single-mode version of the RCPSP. In the multi-mode version of the RCPSP (cf. Talbot, 1982; Patterson et al., 1990) basically each activity $j$, $1 \le j \le J$, can be performed in one out of $M_j$ modes. Each mode is characterized by a specific duration $d_{jm}$ and the resource demand $k_{jmr}$ for each $r \in R$. Once each activity $j$ is assigned one of its $M_j$ modes the schedule classification reduces to the single-mode RCPSP.

## 3. Types of schedules

To start with we consider the JSP where a number $A$ of jobs have to be processed on $M$ machines. Each job consists of $G$ operations, each of which has to be done on one of the $M$ machines. In the classical JSP the number of operations is equal to the number of machines and each operation of a job has to be processed on a different machine.

It can be easily verified that the JSP corresponds to an RCPSP with $|R| = M$ renewable resources, each of which has an availability of one unit per period (cf. Schrage, 1970; Baker, 1974; Stinson et al., 1978; Drexl, 1989).

As already proposed we proceed as follows: Based on the informal definitions given in the

JSP context (Baker, 1974) we extend and formalize them for the RCPSP.

Within the JSP context Baker defines a schedule to be a feasible resolution of resource and logical constraints (Baker, 1974, p. 179). More precisely we define the following:

**Definition 2.** A *schedule* $S$ is a $J$-tuple $S = (ST_1, \ldots, ST_J)$, where $ST_j$ denotes the start time of activity $j$, $1 \le j \le J$.

**Definition 3.** For a given schedule $S$ and a period $t$, $1 \le t \le T$, the set of activities being in progress in period $t$ is $A_t(S)$, where

$$A_t(S) := \{j \mid 1 \le j \le J,\ ST_j + 1 \le t \le ST_j + d_j\}.$$

**Definition 4.** A *schedule* $S$ is called *feasible*, if the precedence relations are maintained, i.e.

$$ST_i + d_i \le ST_j, \quad j = 2, \ldots, J, \quad i \in P_j,$$

and the resource constraints are met, i.e.

$$\sum_{j \in A_t(S)} k_{jr} \le K_{rt}, \quad r \in R, \quad t = 1, \ldots, T.$$

The local or limited left shift for a given schedule $S$ of the JSP is defined as follows (Baker, 1974, p. 181): A local or limited left shift is, "moving an operation block to the left on the Gantt chart while preserving the operation sequences". Since the term 'operation sequence' is not interpretable within the project scheduling context, we define:

**Definition 5** (Wiest, 1964). A *left shift* of activity $j$, $1 \le j \le J$, is an operation on a feasible schedule $S$, which derives a feasible schedule $S'$, such that $ST'_j < ST_j$ and $ST'_i = ST_i$ for $i$, $1 \le i \le J$, $i \ne j$.

**Remark.** If a regular measure of performance $\phi$ is considered and a schedule $S'$ is obtainable from $S$ by a left shift of an activity $j$, $1 \le j \le J$, then $S$ is dominated by $S'$ w.r.t. $\phi$.

**Definition 6.** A left shift of activity $j$, $1 \le j \le J$, is called a *one-period left shift* if we have $ST_j - ST'_j = 1$.

**Definition 7.** A *local left shift* of activity $j$, $1 \le j \le J$, is a left shift of activity $j$ which is obtainable by one or more successively applied one-period left shifts of activity $j$.

**Remark.** Within a local left shift each intermediate derived schedule has to be feasible by definition.

Regarding a schedule where no further local left-shifts are possible Baker defines a global left shift, as to start an operation earlier without delaying any other operation (cf. Baker, 1974, p. 183). Instead we state:

**Definition 8.** A *global left shift* of activity $j$, $1 \le j \le J$, is a left shift of activity $j$ which is not obtainable by a local left shift.

**Remarks.** (a) A global left shift of activity $j$, $1 \le j \le J$, induces $ST_j - ST'_j > 1$.

(b) If a feasible schedule $S'$ is derived from the feasible schedule $S$ by a global left shift, then $S'$ is not obtainable from $S$ by a local left shift, since at least one intermediate schedule is not feasible with respect to the resource constraints.

Based on the notion of a local left shift Baker defines the set of semi-active schedules, to be those schedules in which no local left shift is possible (Baker, 1974, p. 181). By employing our definition of a local left shift (Definition 7), we define:

**Definition 9.** A *semi-active schedule* is a feasible schedule where none of the activities $j$, $1 \le j \le J$, can be locally left shifted.

**Remark.** A feasible schedule can be transformed into a semi-active schedule by a series of local left shifts. Note that the derived semi-active schedule is in general not unique.

Obviously our definition coincides with the definition given by Baker. The remark "In a semi-active schedule the start time of a particular operation is constrained by the processing of a different job on the same machine or by the

processing of the directly preceding operation on a different machine" (Baker, 1974, p. 183) has to be generalized in the following way:

**Remark.** In a semi-active schedule $S$ the start time $ST_j$ of any activity $j$, $1 \le j \le J$, cannot be reduced by one period, because there is at least one resource $r, r \in R$, for which the left over capacity in period $ST_j - 1$ is exceeded by the requirements of activity $j$ or at least one predecessor of activity $j$ is not finished up to the end of period $ST_j - 1$.

For the JSP the set of active schedules is defined as "the set of all schedules in which no global left shift can be made" (Baker, 1974, p. 183). For the RCPSP we use the following generalization:

**Definition 10.** An *active schedule* is a feasible schedule where none of the activites $j$, $1 \le j \le J$, can be locally or globally left shifted.

Finally, in the JSP context (cf. Baker, 1974, p. 185), a non-delay schedule is a schedule where "no machine is kept idle at a time when it could begin processing some operation". Employing the following remark we can give the more general definition of a non-delay schedule.

**Remark.** Each RCPSP can be uniquely transformed into a unit-time-duration RCPSP (UTDRCPSP) where each activity $j$, $1 < j < J$, is split into $d_j$ activities, each of which with duration one (cf. Davis and Heidorn, 1971; Demeulemeester and Herroelen, 1992b). Thus, a feasible schedule $S$ of the RCPSP uniquely corresponds to a feasible schedule *UTDS* of the UTDRCPSP.

**Definition 11.** A feasible schedule $S$ for the RCPSP is called a *non-delay schedule* if the corresponding schedule *UTDS* is active.

By definition we can state the following theorem:

**Theorem.** *Let S denote the set of schedules, **FS** the set of feasible schedules, **SAS** the set of semi-active schedules, **AS** the set of active schedules,* and ***NDS** the set of non-delay schedules. Then the following holds:*

$$NDS \subseteq AS \subseteq SAS \subseteq FS \subseteq S.$$

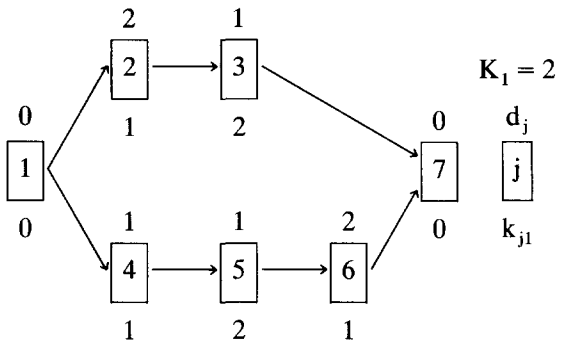


Fig. 2. An example for the RCPSP.

## 4. Examples and illustrations

In order to illustrate the given definitions we consider the example provided in Fig. 2 with $|R| = 1$.

A feasible schedule of the above problem is depicted by the Gantt chart in Fig. 3.

By performing a local left shift (constituting of a one-period left shift) of activities 2 and 3, respectively, and a local left shift (constituting of two one-period left shifts) of activity 6, the semi-active schedule displayed in Fig. 4 is derived. Note that after the first one-period left shift of activity 6 the intermediate schedule $S = (0,2,4,0,1,6,8)$ is feasible.

Regarding the semi-active schedule, clearly none of the activities can be locally left shifted anymore. Nevertheless, activity 6 can be globally left shifted by performing a three-period left shift. Doing so one achieves the active schedule displayed in Fig. 5.

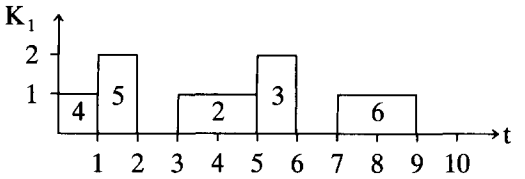


Fig. 3. Feasible schedule for the example problem.

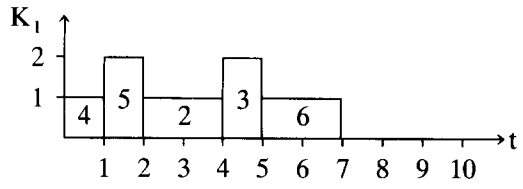


Fig. 4. Semi-active schedule for the example problem.

Again, note that the two intermediate schedules $S' = (0,2,4,0,1,4,6)$ and $S'' = (0,2,4,0,1,3,5)$ are not feasible and thus we have not performed a local left shift. Since none of the activities can be locally or globally left shifted, the schedule is active. Furthermore, the schedule is the unique optimal schedule of the example problem. Optimality can easily be verified by applying the resource-based lower bound as presented in Stinson et al. (1978), the uniqueness can be shown by performing an explicit enumeration with one of the schemes presented in Stinson et al. (1978) or Demeulemeester and Herroelen (1992a).

In order to see whether the optimal solution is a non-delay schedule or not, we transform the example problem into the corresponding UTDRCPSP, where each activity $j$, $1 \le j \le J$, of the RCPSP is transformed into the activites $j1, \ldots, jd_j$ (cf. Fig. 6).

The corresponding solution is displayed in Fig. 7. For this schedule activity 21 can be globally left shifted ($ST_{21} = 2 \rightarrow ST'_{21} = 0$), therefore the schedule for the RCPSP does not belong to the set of non-delay schedules.

Since the optimal schedule is unique, we can state the following: When considering a regular measure of performance, the set of non-delay schedules might not contain an optimal schedule.

The schedule presented in Fig. 8 is the schedule we obtain by assigning activity 2 the start time of 0, which alters the start times of the activites 3,

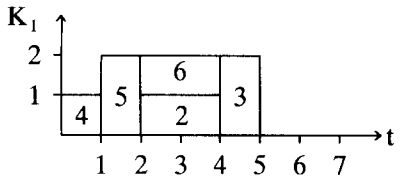


Fig. 5. Active (and unique optimal) schedule for the example problem.

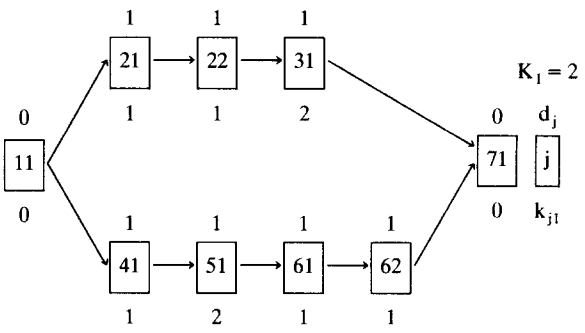


Fig. 6. Corresponding UTDRCPSP.

5 and 6. Again by applying Definition 11, it can be shown that this is a non-delay schedule. Note, since the optimal solution is unique and not a non-delay schedule, we have an empty intersection between the set of optimal solutions and the set of non-delay schedules.

## 5. Implications for disjunctive arc based algorithms

As an illustration of the problems encountered by applying the former schedule classification of the JSP (to the RCPSP) the following examples are given:

A well known extension of the disjunctive arc concept (Balas, 1969) within the JSP context is the approach via partially ordered sets (posets) (Radermacher, 1985/1986; Bartusch et al., 1988) which will now be briefly introduced: An RCPSP is described by the tuple $(A, O_0, x, N, \kappa)$ with the following meaning:

$A$ = $\{\alpha_1, \ldots, \alpha_n\}$: Set of activities.
$O_0$ : Precedence relations.
$x$ = $(x_1, \ldots, x_n) \in R^n_>$ : Duration vector.
$N$ = $\{N_1, \ldots, N_k\}$: A set of $\subseteq$-incomparable (i.e. $\forall\, N_i, N_j \in N$, $i \neq j$, it is $N_i \not\subseteq N_j$) and

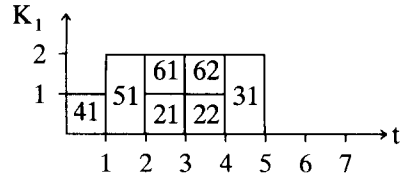


Fig. 7. Corresponding solution for the UTDRCPSP.

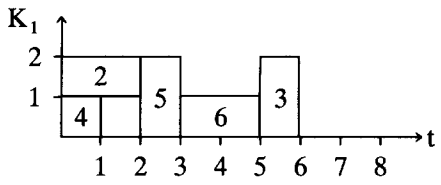


Fig. 8. Non-delay schedule for the Example problem.

$O_0$-independent (i.e. $\forall N_i \in N$ and $\alpha, \beta \in N_i$ is $(\alpha, \beta) \notin O_0$) subsets of $A$, which represent those minimal sets of technologically independent activities that are not allowed w.r.t. resource constraints to be scheduled as a whole at any time.

$\kappa$ : Regular measure of performance

Using Definition 12 the earliest start schedule $ES_{(A, O)}$ of a feasible poset $O$ can be derived.

**Definition 12** (Radermacher, 1985/1986). Let $(A, O_0, x, N, \kappa)$ be a scheduling problem. Call a poset $(A, O)$ feasible if $O_0 \subseteq O$ and no $N_i \in N$ is $O$-independent, i.e. for each $N_i \in N$ there are $\alpha, \beta \in N_i$, $\alpha \neq \beta$, such that $(\alpha, \beta) \in O$.

Due to the minimality of the sets of $N$ and the feasibility of the extensions of the partial order we now have a scheduling problem $(A, O, x, \{\ \}, \kappa)$ where the earliest start schedule $ES_{(A, O)}$ calculated by traditional forward recursion is feasible (with respect to resource availabilities). Nevertheless, after the introduction of the disjunctive arcs the obtained network has to be checked for cycles of positive length in order to meet acyclicity of posets (cf. the JSP-example). For the latter,

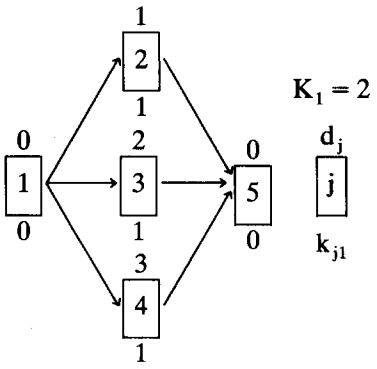


Fig. 9. RCPSP-counterexample.

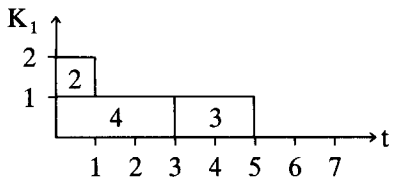


Fig. 10. Earliest start schedule of $(A, O_6)$ for the RCPSP-counterexample.

the resulting earliest start schedule cannot be calculated.

Radermacher (1985/1986, p. 233) states that each earliest start schedule of $ES_{(A, O)}$ with (A, $O$) feasible belongs to the set of semi-active schedules and that the earliest start schedules of $\subseteq$-minimal posets ($\subseteq$-minimal w.r.t. the set of all feasible posets) belong to the set of active schedules. Based on two counterexamples we will show that this generally does not hold for the RCPSP as well as for the JSP.

Consider the RCPSP-example provided in Fig. 9 with $A = \{1, 2, 3, 4, 5\}$, $x = (0, 1, 2, 3, 0)$, $O_0$ as given by the AON and $N = \{N_1 = \{2,3,4\}\}$. Note that the zero duration of dummy source and sink is for convenience of presentation only and is w.l.o.g. not critical w.r.t. the assumptions of the order theoretical approach.

There are 6 $\subseteq$-minimal feasible posets, $O_1 = O_0 \cup \{(2, 3)\}$, $O_2 = O_0 \cup \{(3, 2)\}$, $O_3 = O_0 \cup \{(2, 4)\}$, $O_4 = O_0 \cup \{(4, 2)\}$, $O_5 = O_0 \cup \{(3, 4)\}$, and $O_6 = O_0 \cup \{(4, 3)\}$. Just the earliest start schedules of the first 3 $\subseteq$-minimal feasible posets belong to the set of acitve schedules. Each of the earliest start schedules for the 3 remaining $\subseteq$-minimal feasible posets will yield a feasible schedule only. Fig. 10 shows the feasible schedule resulting for the earliest start schedule of $(A, O_6)$.

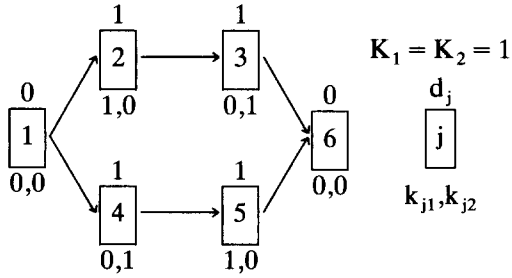


Fig. 11. JSP-counterexample.

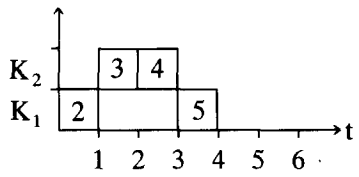


Fig. 12. Earliest start schedule of $(A, O_1)$ for the JSP-counter-example.

Let us further consider the JSP-example provided in Fig. 11 with $A = \{1, 2, 3, 4, 5, 6\}$, $x = (0, 1, 1, 1, 1, 0)$, $O_0$ as given by the AON and $N = \{\{2, 5\}, \{3, 4\}\}$.

There are 4 $\subseteq$-minimal feasible posets $O_1 = O_0 \cup \{(2, 5), (3, 4)\}$, $O_2 = O_0 \cup \{(5, 2), (3, 4)\}$, $O_3 = O_0 \cup \{(2, 5), (4, 3)\}$, and $O_4 = O_0 \cup \{(5, 2), (4, 3)\}$. Just the earliest start schedule of $(A, O_3)$ yields an active schedule, while the earliest start schedule of $(A, O_1)$ (cf. Fig. 12) and $(A, O_4)$ will yield a feasible schedule only. Finally, the early start schedule associated with $(A, O_2)$ is not feasible because it contains a cycle.

## 6. Conclusions

The paper provides a formal definition of semi-active, active, and non-delay schedules for the RCPSP. These schedules establish basic concepts within the job shop scheduling literature. There, they are usually defined in a rather informal way. Using these concepts in the more general RCPSP without giving a formal definition may cause serious problems. Therefore, we formally define semi-active, active, and non-delay schedules for the RCPSP. In addition, we outline some of these problems occurring within the disjunctive arc concept.

The definitions presented above allow deeper insight concerning the performance of exact and heuristic algorithms:

The exact (branch-and-bound based) algorithms for the single-mode case of the RCPSP (cf. Talbot and Patterson, 1978; Stinson et al., 1978; Radermacher, 1985/1986; Christofides et al., 1987; Demeulemeester and Herroelen, 1992a) as well as for the multi-mode case of the RCPSP (cf. Talbot, 1982; Patterson et al., 1989; Sprecher, 1994) are enumerating a subset of the set of feasible schedules. Most noteworthy, the fast procedures (i.e. Stinson et al., 1978, Demeulerneester and Herroelen, 1992a, and Sprecher, 1994) are using the so-called left-shift dominance rule, which checks the feasibility of local left shifts in the partial schedule under consideration. Doing so, branches of the enumeration tree are evaluated and the latter is reduced to the set of semi-active schedules. A question to be raised is, whether further improvement can be achieved by using an extended left-shift dominance rule, which checks the feasibility of global left shifts and hence cuts down the enumeration tree to the set of active schedules.

Also, many heuristic procedures for the single-mode case of the RCPSP generate solutions from a subset of the set of feasible schedules (e.g. the truncated branch-and-bound approach of Alvarez-Valdes and Tamarit (1989), the disjunctive arc based heuristic of Bell and Han (1991), and the local search procedure of Sampson and Weiss (1993)). Again, these approaches can be improved by making use of the local-left shift dominance.

For the two well known heuristic schedule generation schemes, the serial and the parallel approach (Alvarez-Valdes and Tamarit, 1989), it has been shown that the schedules generated belong to the set of active schedules and non-delay schedules, respectively (cf. Kolisch, 1994). Hence, regardless of the priority rule employed, the parallel approach might a priori not be able to generate an optimal solution.

## Acknowledgement

The authors are indebted to Rolf Möhring, Technical University of Berlin, as well as two anonymous referees for helpful suggestions and comments.

## References

Alvarez-Valdes, R., and Tamarit, J.M. (1989), "Heuristic algorithms for resource-constrained project scheduling: A review and an empirical analysis", in: R. Slowinski and J.

Weglarz (eds.), *Advances in Project Scheduling*, Elsevier, Amsterdam, 113–134.

Balas, E. (1969), "Machine sequencing via disjunctive graphs: An implicit enumeration algorithm", *Operations Research* 17, 941–957.

Baker, K.R. (1974), *Introduction to Sequencing and Scheduling*, Wiley, New York.

Bartusch, M., Möhring, R.H., and Radermacher, F.J. (1988), "Scheduling project networks with resource constraints and time windows", *Annals of Operations Research* 16, 201–240.

Bell, C.E., and Han, J. (1991), "A new heuristic solution method in resource-constrained project scheduling", *Naval Research Logistics* 38, 315–331.

Bell, C.E., and Park, K. (1990), "Solving resource-constrained project scheduling problems by $A^*$ search", *Naval Research Logistics* 37, 61–84.

Christofides, N., Alvarez-Valdes, R., and Tamarit, J.M. (1987), "Project scheduling with resource constraints: A branch and bound approach", *European Journal of Operational Research* 29, 262–273.

Conway, R.W., Maxwell, W.L., and Miller L.W. (1967), *Theory of Scheduling*, Addison-Wesley, Reading, MA.

Davis, E.W., and Heidorn, G.E. (1971), "An algorithm for optimal project scheduling under multiple resource constraints", *Management Science* 17, 803–816.

Demeulemeester, E., and Herroelen, W.S. (1992a):, "A branch-and-bound procedure for the multiple resource-constrained project scheduling problem", *Management Science* 38, 1803–1818.

Demeulemeester, E., and Herroelen, W.S. (1992b), "An efficient optimal solution procedure for the preemptive resource-constrained project scheduling problem", Working Paper, Departement of Applied Economic Sciences, Katholieke Universiteit Leuven, Belgium.

Drexl, A. (1990), "Fließbandaustaktung, Maschinenbelegung und Kapazitätsplanung in Netzwerken", *Zeitschrift für Betriebswirtschaft* Jg. 60, 53–70.

Elmaghraby, S.E. (1977), *Project Planning and Control by Network Models*, Wiley, New York.

French, S. (1982), *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop*, Wiley, New York.

Gonguet, L. (1969), "Comparison of three heuristic procedures for allocating resources and producing schedule", in: H.J.M. Lombaers (ed.), *Project Planning by Network Analysis*, North-Holland, Amsterdam, 249–255.

Herroelen, W.S., and Demeulemeester, E. (1992), "Recent advances in branch-and-bound procedures for resource-constrained project scheduling problems", Working Paper, Departement of Applied Economic Sciences, Katholieke Universiteit Leuven, Belgium.

Kolisch, R. (1994), *Project Scheduling under Resource Constraints – Efficient Heuristics for Several Problem Classes*, Physica, Heidelberg, to appear.

Patterson, J.H., Slowinski, R., Talbot, F.B. and Weglarz, J. (1989), "An algorithm for a general class of precedence and resource constrained scheduling problems", in: R. Slowinski and J. Weglarz (eds.), *Advances in Project Scheduling*, Elsevier, Amsterdam, 3–28.

Patterson, J.H., Slowinski, R., Talbot, F.B., and Weglarz, J. (1990), "Computational experience with a backtracking algorithm for solving a general class of precedence and resource-constrained scheduling problems", *European Journal of Operational Research* 49, 68–79.

Pritsker, A.A.B., Watters, L.J., and Wolfe, P.M. (1969), "Multiproject scheduling with limited resources: A zero–one programming approach", *Management Science* 16, 93–107.

Radermacher, F.J. (1985/1986), "Scheduling of project networks", *Annals of Operations Research* 4, 227–252.

Rinnooy Kan, A.H.G. (1976), *Machine Scheduling Problems: Classification, Complexity and Computations*, Martinus Nijhoff, The Hague.

Sampson, S.E., and Weiss, E.N. (1993), "Local search techniques for the generalized resource constrained project scheduling problem", *Naval Research Logistics* 40, 665–675.

Schrage, L. (1970), "Solving resource-constrained network problems by implicit enumeration – Nonpreemptive case", *Operations Research* 18, 263–278.

Slowinski, R. (1989), "Multiobjective project scheduling under multiple-category resource constraints", in: R. Slowinski and J. Weglarz (eds.), *Advances in Project Scheduling*, Elsevier, Amsterdam, 151–167.

Sprecher, A. (1994), *Resource-Constrained Project Scheduling: Exact Methods for the Multi-Mode Case*, Springer, Berlin.

Stinson, J.P., Davis, E.W. and Khumawala, B.M. (1978), "Multiple resource-constrained scheduling using branch and bound", *AIIE Transactions* 10, 252–259.

Talbot, B. (1982), "Resource-constrained project scheduling with time-resource tradeoffs: The nonpreemptive case", *Management Science* 28, 1197–1210.

Talbot, B., and Patterson, J.H. (1978), "An efficient integer programming algorithm with network cuts for solving resource-constrained scheduling problems", *Management Science* 24, 1163–1174.

Wiest, J.D. (1964), "Some properties of schedules for large projects with limited resources", *Operations Research* 12, 395–418.