dr hab. inż. Joanna Józefowska
dr inż. Rafał Różycki,
prof. zw. dr hab. inż. Jan Węglarz
Instytut Informatyki, Politechnika Poznańska

UNCERTAINTY IN DISCRETE-CONTINUOUS PROJECT SCHEDULING

Abstract. The problem of adjusting the discrete-continuous schedule in case of disturbances is presented. The sources of disturbances are discussed. A heuristic approach to adjusting the schedule to the varying available amount of the continuous resource is presented. A computational experiment is presented to illustrate a general reactive scheduling policy.

1. INTRODUCTION

In the scheduling theory the aim of solving a problem is to find an assignment of resources to activities in time to minimise a chosen criterion. In practice, however it often happens that execution of the schedule is not possible. Perturbations may be caused by fluctuation of resources or unexpected changes in the characteristics of an activity. Finally, some activity may simply be late due to unknown reasons. In such cases a question arises: how to update the schedule in order to ensure its feasibility and optimality. This problem is lately considered in the literature. Herroelen and Leus [1] present literature review and a survey of existing approaches. One of the possible approaches to the problem of practical application of schedules is to create a schedule that remains feasible and optimal (sub-optimal) even in case of unexpected events during execution [6].

In this paper we relate the problem of schedule execution to discrete-continuous project scheduling problems.

In Section 2 the problem of project scheduling with discrete and continuous resources is presented and exact as well as heuristic solution approaches are characterized. In Section 3 the problem of disturbances during execution of deterministic schedules is discussed, while in Section 4 a special case of continuous resource variation is presented. Section 5 describes an approach to constructing reactive schedules when delay or early completion of an activity is observed. This approach is illustrated by a computational experiment presented in Section 6. Conclusions and directions for further research are provided in Section 7.

2. PROBLEM FORMULATION

The problem of discrete-continuous project scheduling is defined as follows [3]. We consider a project consisting of *n* precedence- and resource-constrained, nonpreemptable activities that require renewable resources of two types: discrete and continuous ones. We assume m discrete resources are available and vector $\mathbf{r}_i = [r_{i1}, r_{i2}, \ldots, r_{im}]$, $i = 1,\ldots, n$, determines the (fixed) discrete resource requirements of activity i. The total number of units of the discrete resource j, $j = 1,\ldots, m$, is limited by R_j. The single continuous, renewable resource can be allotted to activities in (arbitrary) amounts from the interval [0, 1]. The amount (unknown in advance) of the continuous resource $u_i(t)$, allotted to activity i at time t determines the processing rate of activity i as it is described by the following equation:

$$\dot{x}_i(t) = \frac{dx_i(t)}{dt} = f_i[u_i(t)],\ x_i(0) = 0,\ x_i(C_i) = \tilde{x}_i \tag{1}$$

where $x_i(t)$ is the state of activity i at time t,

f_i is a continuous, nondecreasing function, where $f_i(0) = 0$,

$u_i(t)$ is the amount of the continuous resource allotted to activity i at time t,

C_i is the completion time (unknown in advance) of activity i,

$\tilde{x}_i$ is the processing demand (final state) of activity i.

State $x_i(t)$ of activity i at time t is an objective measure of work related to the processing of activity i up to time t. It may denote, for example, the number of man-hours already spent on processing of activity i, the number of standard instructions in processing of computer program i, and so on. All activities are available at the start of the process.

The problem is to find a feasible assignment of discrete resources and, simultaneously, a continuous resource allocation which minimise the schedule length, i.e. the project duration, $M = \max\{C_i, i = 1, 2,\ldots, n\}$. The continuous resource allocation is defined by a piece-wise continuous, non-negative vector function $\mathbf{u}^*(t) = [u_1^*(t), u_2^*(t),\ldots,u_n^*(t)]$ whose values $\mathbf{u}^* = [u_1^*, u_2^*,\ldots,u_n^*]$ are (continuous) resource allocations corresponding to M^* — the minimal value of M.

Problems of this type where a set of identical parallel machines was the single discrete resource was defined in [4] and examined for nonpreemptable, independent activities.

Let us divide a feasible schedule (i.e. a solution of a discrete-continuous project scheduling problem) into $p \le n$ intervals of lengths M_k, $k = 1,\ldots, p$, defined by the completion times of the consecutive activities.

Let Z_k denote the combination of activities corresponding to the k-th interval. Thus, a *feasible sequence* S of combinations Z_k, $k = 1,2,\ldots, p$, is associated with each feasible schedule. The feasibility of such a sequence requires that:

1. the number of units of the discrete resource j, $j = 1,\ldots, m$, assigned to all activities in combination Z_k, $k = 1,\ldots, p$, does not exceed R_j, i.e. $\sum_{i \in Z_k} r_{ij} \le R_j$, $j = 1,\ldots, m$, $k = 1,\ldots, p$,

2. each activity appears in at least one combination,
3. precedence constraints between activities are satisfied,
4. nonpreemptability of each activity is guaranteed.

The last condition requires that each activity appeared in exactly one or in consecutive combinations in S.

It has been proved in [4] that for convex processing rate functions f_i, $i = 1, 2,\dots, n$, a makespan optimal schedule is obtained by allotting the total amount of the continuous resource to one activity at a time only. Concluding, in an optimal schedule activities are ordered in a sequence fulfilling the precedence constraints and the total amount of the continuous resource is allotted to each activity. Of course, if we assume $r_{ij} \le R_j$, $i = 1, 2,\dots, n$, $j = 1, 2,\dots, m$, the discrete resource constraints are fulfilled, because only one activity is performed at a time. For concave processing rate functions in a makespan optimal schedule as many activities as possible are performed in parallel (see [4]). In this case for a given feasible sequence S one can find an optimal division of processing demands of activities, $\tilde{x}_i$, $i = 1,2,\dots, n$, among combinations in S, i.e. a division which leads to a minimum length schedule from among all feasible schedules generated by S. To this end a nonlinear programming problem can be formulated in which the sum of the minimum-length intervals (i.e. parts of a feasible schedule) generated by consecutive combinations in S, as functions of the $\{\tilde{x}_{ik}\}_{k\in Z_k}$, where $\tilde{x}_{ik}$ is a part of activity i processed in combination Z_k, is minimized subject to the constraints that each activity has to be completed. For concave processing rates f_i, $i = 1,\dots, n$, it is sufficient to consider feasible schedules in which the resource allocations among activities remain constant in each interval k, $k = 1, 2,\dots, p$. In the sequel we will assume that f_i, $i = 1, 2,\dots, n$, are concave.

Let $M_k^*(\tilde{\mathbf{x}}_k)$ be the minimum length of the part of the schedule generated by $Z_k \in S$, as a function of $\tilde{\mathbf{x}}_k = \{\tilde{x}_{ik}\}_{i\in Z_k}$. Let K_i be the set of all indices of Z_k's such that $i \in Z_k$. The following mathematical programming problem is obtained to find an optimal demand division of activities for a given feasible sequence S.

Problem P

Minimize
$$M\left(\{\tilde{\mathbf{x}}_k\}_{k=1}^{p}\right) = \sum_{k=1}^{p} M_k^*(\tilde{\mathbf{x}}_k) \tag{2}$$

subject to
$$\sum_{k\in K_i} \tilde{x}_{ik} = \tilde{x}_i, \qquad i = 1,2,\dots, n \tag{3}$$

$$\tilde{x}_{ik} \ge 0, \qquad i = 1,2,\dots, n; \quad k \in K_i \tag{4}$$

where $M_k^*(\tilde{\mathbf{x}}_k)$ is calculated as a unique positive root of the equation:

$$\sum_{i\in Z_k} f_i^{-1}\left(\tilde{x}_{ik}/M_k\right)=1 \tag{5}$$

which can be solved analytically for some important cases (see [7]).

Of course, constraints (3) correspond to the condition of fulfilling processing demands of all activities. It was proved by Węglarz in [7] that the objective function (2) is always a convex function. In consequence, our problem is to minimize a convex function subject to linear constraints.

After finding an optimal division $\tilde{x}_{ik}$, $i=1,2,\ldots,n$, $k\in K_i$ of $\tilde{x}_i$'s the corresponding continuous resource allocation for combination Z_k is given as

$$u^*_{ik}=f_i^{-1}\left(\tilde{x}^*_{ik}/M^*_k\right),\ i\in Z_k \tag{6}$$

Thus, an optimal continuous resource allocation for a given feasible sequence can be calculated by solving Problem P. In consequence, a globally optimal schedule can be found by solving the continuous resource allocation problem optimally for all feasible sequences and choosing a schedule with the minimum length. In this sense the process of finding an optimal schedule can be viewed as a search process over the space of all feasible sequences for a given problem instance. Unfortunately, in general, the number of all feasible sequences grows exponentially with the number of activities. Therefore it is justified to apply local search metaheuristics such as Simulated Annealing (SA), Tabu Search (TS) or Genetic Algorithms (GA) operating on a space of all feasible sequences [2].

3. PRACTICAL ASPECTS OF PROJECT SCHEDULING

The methodology presented above assumes complete information about the scheduling problem and a static deterministic environment within which the schedule will be executed. This is a general characteristic of the deterministic approaches to scheduling. A schedule constructed basing on such assumptions will be called a *predictive* one.

However, in practical applications the problem parameters are subject to uncertainty. Quite often it may be difficult to predict the processing demand and processing rate function of the activities. In such situation estimation from analysis of historical data is used to build the predictive schedule. Herroelen and Leus [1] present an extensive survey of approaches to project scheduling under uncertainty. They examine the following methods: reactive scheduling, stochastic project scheduling, stochastic GERT network scheduling, fuzzy project scheduling, robust (proactive) scheduling and sensitivity analysis.

In this paper we would like to exploit some properties of discrete-continuous scheduling in order to create good reactive schedules without changing the discrete part of the predictive schedule and thus saving a lot of computational time.

By a disturbance in a schedule we will understand a situation where a deviation of the completion time of an activity is observed (delayed or early completion). There may be several reasons of disturbances. Let us recall that a problem instance is characterised by the following parameters: $\tilde{x}_i(t)$, $i=1,\ldots,n$, f_i, $i=1,\ldots,n$, r_{ij}, $i=1,\ldots,n$, $j=1,\ldots,m$, R_j, $j=1,\ldots,m$, and U, where U is the total available amount of the continuous resource available at time t. Notice, that while constructing the schedule we have assumed that

$U = 1$, however during the schedule execution this amount may vary. Variation of any of the parameters listed above may cause disturbances in execution of the predictive schedule. If the processing demand $\widetilde{x}_i(t)$ of job i changes, then its processing time also changes. Availability of resources is the most common source if disturbances. It is easy to observe that variation in the available number of units of a discrete resource may not influence the schedule, while any decrease of the available amount of the continuous resource must result in increased processing time of at least one activity. Similarly, resource requirements of an activity may not influence the schedule in case of discrete resources. In case of continuous resources this demand is represented by the processing rate function. A change of the processing rate function usually results in different than expected completion time. On the other hand, however if a critical discrete resource is not available, the delays may be significant.

The practical execution of a project predictive schedule in the presence of disturbances is impossible without some additional assumptions. In discrete-continuous project scheduling we assume that execution of activities meets the following rules:

- the order of the starting times of activities corresponds to the order defined in the predictive schedule;
- the continuous and discrete resource allocation corresponds to the allocation established in the predictive schedule;
- the continuous resource allocation may change only at the completion time of the activity completed next in the predictive schedule;
- if the total number of units of a critical discrete resource or total amount of the continuous resource is insufficient, the execution of all activities is suspended;

The abovementioned rules can be formulated in the form of the following RealExecution procedure:

```
procedure RealExecution;
begin
 set k=1;
 repeat
         basing on the feasible sequence $S$ determine activity $a$ which is going to be
         completed in combination $Z_k$;
         execute the activities from the combination $Z_k$ (allocate the discrete and
         continuous resources according to the predictive schedule);
         until the resource availability of any resource is insufficient suspend the
         execution of all activities;
         wait until activity $a$ is completed;
         set k=k+1;
 until k<p
end
```

4. THE CASE OF THE CONTINUOUS RESOURCE VARIATION

We have mentioned in Section 3 that the variation of the available amount of the continuous resource must influence the processing time of at least one activity. We will show below how to optimally react to this type of disturbances.

Let us consider a predictive schedule of the form presented in Fig. 1.

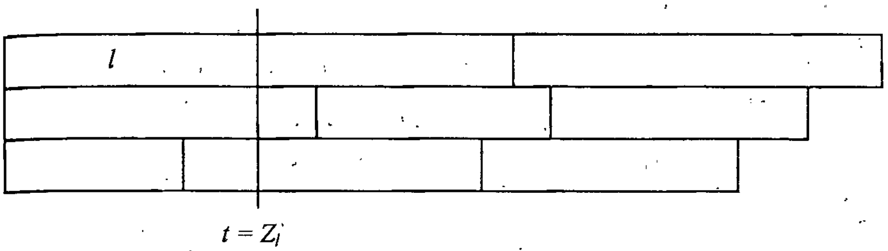

Fig. 1. A predictive schedule

At some point in time, say $t = Z_l$, the available amount of the continuous resource decreases to $U' < 1$. We will show that for identical power processing rate functions $f_i(u_i) = f(u_i) = u_i^{1/\alpha}$, $\alpha \geq 1$in order to obtain an optimal schedule for the remaining activities it is enough to update the assignment of the continuous resource in consecutive intervals, so that: $u'_{ik} = U'u_{ik}$. Let us observe that in each interval corresponding to Z_k, $k > l$ the following equations hold:

$$\sum_{i \in Z_k} \left(\frac{\tilde{x}_{ik}}{M_k} \right)^{\alpha} = 1$$

since we assume that the discrete part of the predictive schedule can not change, we have also

$$\sum_{i \in Z_k} \left(\frac{\tilde{x}_{ik}}{M'_k} \right)^{\alpha} = U'$$

thus

$$\left(M'_k\right)^{\alpha} = \frac{M_k^{\alpha}}{U'} \text{ and finally } u'_{ik} = \left(\frac{\tilde{x}_{ik}}{M'_k} \right)^{\alpha} = \left(\frac{\tilde{x}_{ik}}{M_k} \right)^{\alpha} U' = U' u_{ik} .$$

This statement however is not true for arbitrary processing rate functions. Let us notice that for two jobs with $f_1(u) = u$ and $f_2(u) = \sqrt{u}$ the procedure described above (i.e. $u'_1 = U'u_1$ and $u'_2 = U'u_2$) will result in the following inconsistency:

$$M'_k = \frac{\tilde{x}_1}{U'u_1} = \frac{M_k}{U'} \neq \frac{M_k}{\sqrt{U'}} = \frac{\tilde{x}_2}{\sqrt{U'u_2}}$$

5. THE CASE OF LATE/ EARLY ACTIVITIES

In this section we will describe the method where the continuous resource allocation is modified in response to the late or early completion of activities performed according to the predictive schedule. As it was already mentioned the given feasible sequence of activities will remain unchanged. Otherwise the feasibility of the allocation of the discrete resources could be violated.

The proposed method is dedicated to the case where functions f_i $i = 1, 2, ..., n$, are concave. Moreover, the method bases on the assumption that the present state of an activity is unknown during its execution. Although the proposed model of activity execution (1) allows changing the allocation of the continuous resource to an activity during its execution, we will assume that this allocation may be modified at the completion time of an activity only.

Let S consisting of p combinations represent a feasible sequence for the given instance of the project scheduling problem. Moreover, let us assume that the optimal demand division of activities as well as optimal continuous resource allocation are known for S.

The idea of the method is based on the assumption that any disturbances (late or early completion of an activity) in the execution of the schedule result from an unforeseen change of the processing demand of an activity. The adequate reactive allocation of the continuous resource to the executed activities bases on periodically modified information on the processing demand of the portion of these activities that is not yet completed. To this end the actual processing demands of the executed activities are measured and compared with the processing demands assumed in the predictive schedule.

The method is described in details in the following ReactiveExecution procedure:

```
procedure ReactiveExecution;
begin
 set k:=1;
 repeat
          basing on the feasible sequence S determine activity a which is going to be
          finished in combination $Z_k$;
          if any value of $\mathcal{X}_{ik}$ ($i \in Z_k$) differs from the value from predictive schedule,
          compute a new allocation of the continuous resource using (5) and (6);
          execute the activities from the combination $Z_k$ (allocate the discrete resources
          according to the predictive schedule; the continuous resource allocate to
          activities in computed amounts;);
          until the resource availability of any resource is insufficient suspend the
          execution of all activities;
          wait until any activity is completed; {let b is the completed activity}
          compute the actual execution time and completed portion $real\mathcal{X}_{ik}$ ($i \in Z_k$) of
          executed activities;
           if (a=b) for each activity i ($i \in Z_{k+1}$) set $\mathcal{X}_{ik+1} := \mathcal{X}_{ik+1} + real\mathcal{X}_{ik} - \mathcal{X}_{ik}$;
          else begin
```

```
            for each combination $Z_l$ ($l : l \in K_b \wedge l \geq k$) set $\tilde{x}_{bl} := 0$
            determine max $s$ ($s: b \in Z_s$);
            if ($s<p$) for each $i$ ($i \in Z_{s+1} \wedge i <> b$) $\tilde{x}_{is+1} := \tilde{x}_{is+1} + \tilde{x}_{is}$ ;
            for $l:=s$ downto $k+1$
                begin
                    set $Z_l := Z_{l-1}$
                    for each activity $i$ ($i \in Z_l$) set $\tilde{x}_{il} := \tilde{x}_{il-1}$;
                end;
            if ($real\tilde{x}_{ak} < \tilde{x}_{ak}$)
                    for each activity $i$ ($i \in Z_{k+1}$) set $\tilde{x}_{ik+1} := \tilde{x}_{ik+1} + real\tilde{x}_{ik} - \tilde{x}_{ik}$;
        end;
    set k:=k+1;
until k<p
end.
```

Notice, that in most practical situations the above procedure will calculate the new continuous resource allocation after the completion of each activity. Thus such a strongly reactive approach is applicable only to those cases where the modified continuous resource allocation can be found analytically.

6. COMPUTATIONAL EXPERIMENTS

In the computational experiments we have analysed the most general and difficult case where the processing time of each activity is uncertain. We assume that set of R machines is the only discrete resource and the continuous resource is available in amount $U = 1$. The processing rate function for each activity f_i has the form $f_i = u_i^{1/2}$, $i = 1, 2, ..., n$ and processing demands $\tilde{x}_i$, $i = 1, 2, ..., n$ were generated randomly with the uniform distribution from the interval [1..100]. To simulate the uncertainty in the discrete-continuous projects we generated the actual processing demand of each job with the normal distribution $N(\tilde{x}_i, coef \cdot \tilde{x}_i)$, where *coef* is a parameter which allows controlling the value of the standard deviation (*coef* > 0). The other parameters of the problem remain unchanged. We have tested various combinations of problem sizes ($n \in \{20, 30\}$, $R \in \{2, 3, 5, 10\}$) and values of the coefficient *coef* (*coef* $\in \{0.1, 0.2, 0.3\}$). For each combination of the input data 100 instances of the problem were generated randomly.

The value of the makespan (ReactC_{max}) obtained using ReactiveExecution procedure was compared with the corresponding value of the schedule length (RealC_{max}) given by the RealExecution procedure and with the makespan (SolverC_{max}) obtained by optimal resource allocation (the related convex mathematical programming problem was solved by CFSQP solver [5]). In Table 1. the average relative deviation from the SolverC_{max} (avgRD) as well as the maximum relative deviation from the SolverC_{max} (maxRD) are presented. In column Worse the number of instances where ReactC$_{max}$ > RealC$_{max}$ is showed.

Problem size		RealC$_{max}$ (avgRD/maxRD)	ReactC$_{max}$ (avgRD/maxRD)	Worse (#)
n=20, *R*=2	*Coef*=0.1	0.028/0.097	0.014/0.049	17
	Coef=0.2	0.061/0.189	0.033/0.090	22
	Coef=0.3	0.088/0.234	0.049/0.128	22
n=20, *R*=3	*Coef*=0.1	0.050/0.176	0.027/0.166	21
	Coef=0.2	0.108/0.445	0.056/0.314	15
	Coef=0.3	0.157/0.690	0.073/0.148	15
n=20, *R*=5	*Coef*=0.1	0.083/0.271	0.043/0.139	20
	Coef=0.2	0.172/0.543	0.074/0.202	7
	Coef=0.3	0.254/0.824	0.100/0.232	2
n=30, *R*=2	*Coef*=0.1	0.029/0.087	0.014/0.060	21
	Coef=0.2	0.067/0.197	0.033/0.080	19
	Coef=0.3	0.105/0.286	0.052/0.134	14
n=30, *R*=3	*Coef*=0.1	0.048/0.133	0.027/0.074	19
	Coef=0.2	0.103/0.269	0.057/0.192	12
	Coef=0.3	0.153/0.397	0.079/0.185	9
n=30, *R*=5	*Coef*=0.1	0.074/0.201	0.046/0.175	19
	Coef=0.2	0.155/0.423	0.089/0.190	17
	Coef=0.3	0.230/0.638	0.114/0.221	10
n=30, *R*=10	*Coef*=0.1	0.121/0.442	0.060/0.231	9
	Coef=0.2	0.245/0.866	0.106/0.281	5
	Coef=0.3	0.352/1.254	0.135/0.291	1

Table 1. Results of the computational experiments

The presented results show that in most cases the proposed reactive continuous resource allocation generates better schedules than the procedure based on the predictive schedule only. As it was easy to predict, the bigger coefficient *coef* the worse the resulting schedule for both execution procedures compared. Surprisingly in the highly unpredictable environment of the tests the instances where the RealC$_{max}$ < ReactC$_{max}$ are quite rare.

7. CONCLUSIONS

The proposed heuristic generates results satisfactory from the practical point of view. The maximum relative deviation from the schedule with optimum resource allocation may exceed 30%, but simple realisation of the predictive schedule may result in much worse schedules (125% deviation from optimum). Although in some instances the

$RealC_{max}$ is better than $ReactC_{max}$, this happens rarely and on average the heuristic outperforms the RealExecution procedure.

Further research will be carried out to extend the proposed approach to other sources of schedule disturbances. The method will be generalised to address problems with disturbances of continuous as well as discrete resources.

ACKNOWLEDGEMENT

This research was supported by a research grant of the Polish State Committee for Scientific Research No. 4T11F 001 22.

LITERATURE

1. Herroelen W., R. Leus, Project scheduling under uncertainty, invited paper to be published in the special issue of EJOR containing selected papers from PMS 2002.
2. Józefowska J., Mika M., Różycki R., Waligóra G., Węglarz J., *Local search metaheuristics for discrete-continuous scheduling,* European Journal of Operational Research, 107/2, 1998, 354-370.
3. Józefowska J., M. Mika, R. Różycki, G. Waligóra, J. Węglarz, *Project scheduling under discrete and continuous resources*, w: J. Węglarz (Ed.), Project Scheduling: Recent Models, Algorithms and Applications, Kluwer, Dordrecht, 1999, 289-308.
4. Józefowska J., Węglarz J., *On a methodology for discrete-continuous scheduling,* European Journal of Operational Research, 107/2, 1998, 338-353.
5. Lawrence C.,Zhou J.L.,Tits A.L. 1995. *Users guide for CFSQP ver.2.3.*Available by e-mail:andre@eng.umd.edu
6. Sevaux M., K. Sörensen, A genetic algorithm for robust schedules in a just-in-time environment, Research Report LAMIH/SP-2003-1, University of Valenciennes, December 2002.
7. Węglarz J., Modelling and control of dynamic resource allocation project scheduling systems. In: S. G. Tzafestas (ed.), Optimization and Control of Dynamic Operational Research Models. North-Holland, Amsterdam 1982, 105-140.