Marcin Klimek

RESOURCE ALLOCATION ALGORITHMS FOR ROBUST PROJECT SCHEDULING

Key words: resource allocation, robust project scheduling, resource constraints

Introduction

In recent years, Make-To-Order systems (*MTO*) have been increasingly often applied in production, particularly in manufacturing of custom products for an individual customer, whose requirements are variable and diverse. Each manufacturing order in *MTO* is treated as a separate project, created in consultation with the customer. Projects (undertakings) are carried out, *e.g.* in such domains as research and development, public works, construction works, IT, etc.

Together with the growth of interest in carrying out manufacturing processes according to a project, the need for research concerning project scheduling has emerged. The present study analyses a very useful practical problem of scheduling the project with limited availability of resources, i.e. *RCPSP* (*Resource-Constrained Project Scheduling Problem*).

Project execution is accompanied by uncertainty, related, *e.g.* to the variability of customer's requirements, to difficulty in assessing duration of tasks (particularly of unique or inventive activities), to temporary unavailability of resources (machine failures), *etc.* The effect of interferences in task scheduling can be eliminated, among others, by robust scheduling. A robust schedule is created during the project planning phase, before the project implementation start. It is defined as task scheduling which, due to its properties, is resistant to disruptions that may emerge during production (Jensen 2001). Robustness is also understood as the ability of the schedule to compensate for the effects of insignificant increases in activity durations which may be caused by uncontrolled factors (Al-Fawzan, Haouari 2005). A similar approach is applied in this paper.

The task of robust scheduling is to anticipate productive disruptions in order to minimize the differences between the scheduled completion time and the actual time of project completion (*stability of makespan* approach) and/or in order to minimize changes (*e.g.* concerning task start times or assignment of resources to activities) in a planned schedule during its execution (*solution robustness* approach).

The RCPSP problem involves two stages of optimization, intended to create a schedule that would be most resistant to disruptions:

1. *robust resource allocation* – proper assignment of resources for execution of individual activities,
2. *robust buffer allocation* – inserting time and resource buffers; buffer allocation usually takes place with established assignment of resources to tasks (Leus 2003) (Leus *et al.* 2004).

This paper presents issues related to allocation of resources to activities and to the effect of this allocation on schedule robustness. It describes a mathematical model of resource allocation, for which robustness metrics – those already known as well as those proposed by the author – are presented.

Formulating the problem

Projects in the *RCPSP* problem are presented in the *AON* network (*Activity On Node*) as an acyclic, uniform, simple directed graph *G(V, E)*, in which *V* is a set of nodes corresponding to activities (tasks, operations), and *E* is a set of arcs that describe order relations between tasks. Set *V* consists of *n* actions, numbered from 1 to *n* in a topological order, *i.e.* the predecessor has a lower number than the successor. Two dummy tasks are included into the project, 0 and n+1, of zero duration ($d_0 = d_{n+1} = 0$) and of zero demand for resources, representing, respectively, the initial summit and the final summit of *G(V, E)* graph.

Relations occurring between the tasks are of the *finish-start, zero-lag precedence* type, in which a successor can begin immediately after the end of a predecessor:

$$s_i + d_i \leq s_j, \qquad \forall (i,j) \in E \quad , \tag{1}$$

where:

s_i – time of starting activity *i*,

d_i – time of executing activity *i*.

Task execution requires resources. Their amount is limited, i.e. at any moment of time *t*, consumption of resources does not exceed available amounts:

$$\sum_{i \in A(t)} r_{ik} \leq a_k, \qquad \forall t, \forall k \quad , \tag{2}$$

where:

A(t) – set of tasks performed within the time span *[t*-1, *t]*,

a_k – amount of available k type resources ($k = 1,\ldots, K$; where K – number of resource types),

r_{ik} – demand of activity i for type k resource.

Resources are *renewable* i.e. their amount is fixed (equal to a_k for each $k = 1,\ldots, K$), regardless the load in previous periods.

Before allocating resources, a nominal schedule is created, which consists in finding times of individual task starts $s_0, s_1, \ldots s_{n+1}$, while satisfying the above specified order and resource constrains. The most frequent optimisation criterion is *makespan* minimization. It involves searching for a schedule of a minimum time to complete the entire project (the time of project execution is equal to the time of beginning a final dummy activity s_{n+1}).

A given nominal schedule can be carried out with many various allocations of resources. Establishing allocation of resources to tasks is a significant issue from the point of view of robustness of a planned schedule. The problem of resource allocation for the *RCPSP* problem has been a subject of many research studies (Leus 2003) (Leus *et al.* 2004) (Deblaere *et al.* 2006) (Policella 2005). It is a strongly *NP*-hard task, even with one type of resources (Leus 2003).

Description of the resource allocation problem applies the notion of *resource flow networks* (Artigues *et al.* 2003). This is a $G(V, E \cup E_R)$ network, containing nodes from the original network of project activities $G(V, E)$ and extra arcs (E_R set) that join all pairs of nodes (activities) between which flows of resources $f(i, j, k)$ (natural numbers) occur for each type of resource k, from activity i coming to an end to activity j which is beginning. The E_R set contains only those arcs which do not occur in original network $G(V, E)$. Constraints for the resource allocation problem can be formulated in the following way (Leus 2003):

- for each type of resources from set K, the sum of all resources of a given type, going out of a dummy start activity is equal to the sum of those resources going into a dummy end activity, and amounts to a_k:

$$\sum_{j \in V} f(0, j, k) = \sum_{j \in V} f(j, n+1, k) = a_k \qquad \forall k \in K \quad , \quad (3)$$

- for each type of k resources, a sum of all nodes of a given type going into a given node representing a non-dummy activity is equal to the sum of those resources going out of this node, and amounts to r_{ik}:

$$\sum_{j \in V} f(i, j, k) = \sum_{j \in V} f(j, i, k) = r_{ik} \qquad \forall i \in V \setminus \{0, n+1\}, \forall k \in K \quad ,(4)$$

A given schedule may be realized by various resource flow networks, which may differ in terms of resistance to production disruptions occurring during the production time.

Robust resource allocation

While analysing robust resource allocation, it should be remembered that each increase in the duration of the predecessor results in delaying the start of the successor if the start time of the successor is equal to the end time of the predecessor. Each extra arc in the E_R set is a new precedence constraint, which reduces the robustness of the schedule. Consequently, the problem of resource allocation optimization amounts to the issue of minimizing extra arcs (Policella 2005) (Deblaere *et al.* 2006). Robust allocation algorithms assign activities with precedence relations to be realized by the same resources (Policella 2005) and maximize sums of flows between individual tasks (Leus 2003). The problem of minimizing the number of arcs on the activity network is also solved by integer programming (Deblaere *et al.* 2006).

In order to reduce the span of search, so-called *unavoidable arcs* are looked for at the beginning. Activities i and j are linked with an (unavoidable) arc in the resource flow network if it is forced by the schedule subject to resource allocation. Such a situation occurs when the number of available resources of a given type k (without taking into consideration those resources that realized activity i), at the moment of task start j ($t = s_j$), is lower than the demand of activity j for resources. It can be formally expressed in the following way (Deblaere *et al.* 2006):

$$(i,j) \in E_U \Leftrightarrow \exists k : a_k - \sum_{l \in J_t} r_{lk} - \max(0, r_{ik} - \sum_{m \in M_t} r_{mk}) < r_{jk}$$

$$\forall i,j \in V : s_j \geq s_i + d_i, \quad (i,j) \notin E, \tag{5}$$

where:

E_U – set of inevitable arcs ($E_U \subset E_R$),

J_t – set of tasks which are being executed at the moment of $t = s_j$,

M_t – set of such tasks m, for which start times satisfy inequalities $s_j > s_m \geq s_i + d_i$, which at the moment of $t = s_j$ use resources assigned to activity i.

Table 1 presents information on a model project which will be used in robust resource analysis.

Tab. 1. Information on tasks in a model project with one resource (K=1).

Activity i	Duration d_i	Demand for resource r_i	Direct successors of activity i
0	0	0	1, 2, 3, 5
1	5	9	9
2	2	4	6
3	2	6	4
4	2	3	7
5	3	8	8
6	2	5	8
7	3	2	8
8	2	1	9
9	0	0	none

Fig. 1 presents a schedule with model allocation of resources for the project described in Tab. 1.

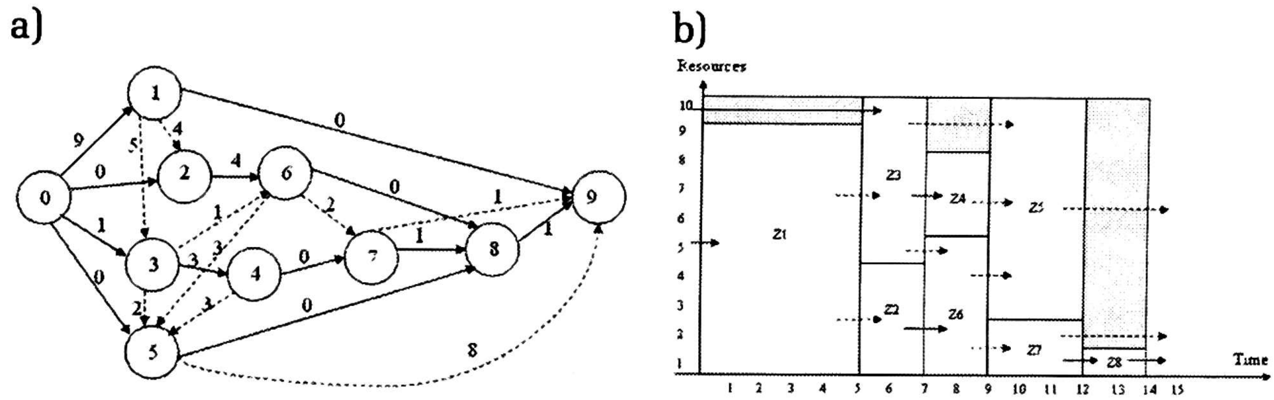

Fig. 1. a) Resource flow network for a model project consisting of eight tasks with indicated (next to the arrows) resource flows $f(i,j)$. Dashed arrows indicate extra arcs, b) Schedule with allocation of resources (with resource flow points indicated by arrows).

Extra arcs (forming the E_R set) are indicated in Figures 1a and 1b with dashed arrows in the resource flow network. For resource flow network in Figure 1a, the E_R set is made of 9 arcs (1.2), (1.3), (3.5), (3.6), (4.5), (5.9), (6.5), (6.7), (7.9). They include the following unavoidable arcs: (1.2), (1.3), (3.6), (4.5), (5.9), (6.5), (7.9). Allocation of resources presented in Figure 1 does not contain a minimum number of arcs. Arcs (3.5) and (6.7) are not unavoidable and can be theoretically removed from the E_R set for different allocation of resources. Elimination of arc (3.5) leads to the creation of an extra arc (3.7), *i.e.* it will not reduce the number of elements in the E_R set. On the other hand, removal of arc (6.7) is possible; therefore a minimum number of elements in the E_R set is 8.

Fig. 2a and 2b present a model resource flow network with a minimum number of extra arcs.

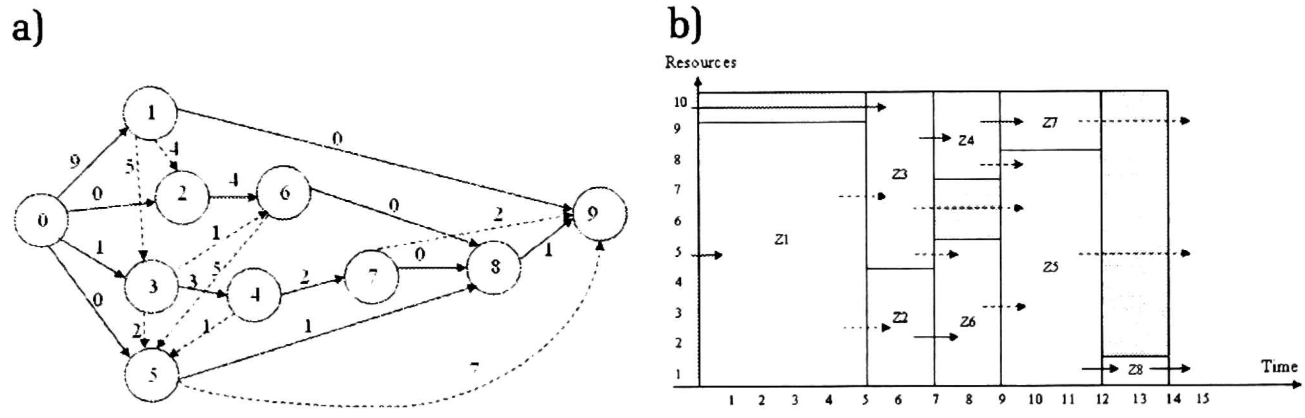

Fig. 2. a) Resource flow network,
b) Gantt's schedule with a minimum number of extra arcs.

The lack of resource flow between activities in a direct precedence relation is often an unfavourable solution. In practice, order relations often result from technological conditions. When the same resources work on subsequent activities from the project network, it is easier to account for and to control the progress of the production process. Therefore, resources should be transferred in the first place from the predecessor to the direct successor.

The next chapter presents robustness measures for the robust resource allocation problem. They take into account rules of robust resources allocation described here.

Measures of resource allocation robustness

Assessment of the robustness of resource allocation involves analysing of the resource flow network and strives to minimize changes occurring in the resource flow network $G(V, E \cup E_R)$, as compared to the original network of activities $G(V, E)$. *Flexibility* (*flex*) is used as a measure of robustness for resource allocation (Policella 2005). The value of *flex* depends on the number of pairs of activities in the resource flow network $G(V, E \cup E_R)$, between which no order dependencies occur. For the *RCPSP* problem considered, in which a topological order of tasks and precedence relations of the finish-start, zero-lag precedence occur, *flex* can be calculated as follows:

$$\text{flex} = 1 - \frac{\#(E \cup E_R)}{\frac{n \cdot (n-1)}{2}}, \quad (6)$$

where:

$\frac{n \cdot (n-1)}{2}$ – number of all possible pairs of activities in the project,

$\#(E \cup E_R)$ – number of arcs in the resource flow network (including extra arcs).

Flexibility indicates what part of tasks is not related by precedence relations. A higher value of *flex* means a lower degree of dependency between tasks, and greater robustness of the schedule. The lack of relationship between two activities is favourable, since a possible delay in ending one activity does not effect the moment of beginning another activity. The problem of maximisation of *flex* indicator is equivalent to the issue of minimizing the number of extra arcs.

Flexibility is a simple measure, but it does not take into account many aspects related to the robustness of scheduling. For a given nominal schedule, there often exist many allocations of resources of the identical value of *flex*, which are more or less susceptible to production disturbances. The *flex* indicator does not allow for, *e.g.* susceptibility to disturbances of individual extra arcs. The author emphasizes that some arcs n the E_R set have lower influence on ranking robustness. These are arcs *(i, j)*, linking activity *i* with activity *j* in situation when resources flow between activity *i* and activity *j*, but those resources are unused for some time. For the example in Fig. 1, this is an arc (3.5), which uses resources 9, 10 that not used in 7-9 time span from (the area filled in grey in Fig. 1b).

A more advanced approach to measure robustness is to establish the effect of prolonging individual activities on production stability. A function of resource allocation becomes then an objective function of reactive scheduling – measurement of stability of a completed schedule. Such an approach is applied, among others, to the *RCPSP* problem with minimization of weighted costs of instability (Deblaere *et al.* 2006). Weighted cost of instability is established experimentally by applying various courses of production, generated on the basis of the statistical knowledge concerning durations of tasks. In each course, task durations are taken at random, each from the β schedule of parameters specified in the experiment.

It has been assumed in this paper that there is no statistical knowledge concerning the variability of tasks duration, *i.e.* prolonging the scheduled duration of each of the tasks is equally probable. Activities are prolonged for reasons that are impossible to establish during the phase of planning, *i.e.*

errors in estimating duration, unfavourable weather conditions, failures, etc. Objective functions for resource allocation are proposed below. They take into account the influence of prolonging individual tasks on production stability, *i.e.* the scope of changes in completed scheduling in relation to the planned one.

A robustness measure proposed by the author is function F_1, calculated according to formula (7) as a sum of time delays in the start of all tasks in relation to the schedule, with the assumption that each of the tasks is prolonged by one time unit.

$$F_1 = \sum_{i=1}^{n+1} (s_i^{all} - s_i), \tag{7}$$

where:

s_i^{all} – rescheduled time of task i start while prolonging each of the activities by 1.

In order to establish F_1, a modified schedule is created, with the resource flow network $G(V, E \cup E_R)$, taking into consideration changed durations of tasks (for each task $j = 1 \ldots n$ duration equals $d_j + 1$). The simplest version of this measure can be the duration of the entire project for a modified schedule, equal to s_{n+1}^{all}.

Another proposed indicator of robustness is function F_2, calculated pursuant to the formula (8) as a sum of the number of rescheduled tasks resulting from delaying each of the activities by one time unit.

$$F_2 = \sum_{j=1}^{n} \left(\sum_{i=1}^{n+1} (s_i^j - s_i) \right), \tag{8}$$

where:

s_i^j – time of task i start while prolonging activity j by 1,

$s_i^j = s_i, \quad \forall (i,j) : s_i + d_i \geq s_j$ occur.

The number of delayed tasks (rescheduled by one time unit) while prolonging a given task by 1 is determined with the assumption that other tasks are realized according to the plan. Time shifts are calculated on the basis of the resource flow network $G(V, E \cup E_R)$ analysed.

The schedule of resource allocation with minimal value of F_1 or F_2 is resistant to disturbances. With such an allocation, possible insignificant prolonging of activities reveals the lowest possible effect on the stability of executed scheduling (delay of other tasks). Measures of F_1 and F_2 make it possible to analyse those properties of schedules that may not be considered in the *flex* indicator.

Summary

This article presents the issue of resource allocation for a Resource-Constrained Project Scheduling Problem. The focus is on determining rules for robust allocation of resources. The article defines criteria of assessing a resource flow network that can be more useful than robustness measures applied so far in research.

The subject of further research will be, *e.g.* development of efficient algorithms of robust resource allocation for proposed measures of robustness.

Abstract

The article presents the problem of robust allocation of resources in a Resource-Constrained Project Scheduling Problem. It discusses the importance of robust scheduling for execution of actual projects.

The issue of resource allocation is analysed for a model project, with indication of allocation principles, which may influence the increase in robustness of the obtained resource allocation network. The author also proposes measures for resource allocation robustness that take into consideration the influence of prolonging individual tasks on the stability of the schedule completed. Those measures make it possible to carry out a more precise analysis concerning the properties of the schedule with resource allocation in terms of its susceptibility to disruption in comparison to a flexibility indicator, *flex*, that has been applied so far in research.

References

1. Al-Fawzan M., Haouari M., A bi-objective problem for robust resource-constrained project scheduling, International Journal of Production Economics, 96, 2005, pp.175-187.
2. Artigues C., Michelon P., Reusser S., Insertion techniques for static and dynamic resource-constrained project scheduling, European Journal of Operational Research 149(2), 2003, pp. 249-267.
3. Deblaere F., Demeulemeester E.L., Herroelen W.S., Van De Vonder S., Proactive resource allocation heuristics for robust project scheduling, Research Report KBI_0608, K.U.Leuven, 2006.

4. Jensen M.T., Improving robustness and flexibility of tardiness and total flow-time job shops using robustness measures. Applied Soft Computing 1, 2001, pp.35-52.
5. Leus, R., The generation of stable project plans. Ph.D. dissertation, Katholieke Universiteit Leuven, Belgium, 2003.
6. Leus R, Herroelen W., Stability and resource allocation in project planning, IIE Transactions, 36(7), 2004, pp.667-682.
7. Policella N., Scheduling with Uncertainty – A Proactive Approach using Partial Order Schedules, Ph.D. dissertation, La Sapienza University, Rome, 2005.