BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 59, No. 1, 2011
DOI: 10.2478/v10175-011-0008-z

Resource allocation for robust project scheduling

M. KLIMEK[1]* and P. ŁEBKOWSKI[2]

[1] Institute of Informatics Pope John Paul II State School of Higher Education, 95-97 Sidorska St., 21-500 Biała Podlaska, Poland
[2] Department of Operations Research and Information Technology, AGH University of Science and Technology, 10 Gramatyka St., 30-067 Kraków, Poland

Abstract. The paper presents a formal description of the resource allocation problem for resource-constrained project scheduling. A schedule can be executed by various resource flow networks which may differ in terms of resistance to disruptions occurring during project execution. The authors define such criteria of evaluating a resource flow network which can be more useful than robustness metrics used so far in the research. The authors discuss the importance of robust scheduling for execution projects and propose metrics for resource allocation robustness which take into consideration the stability of the final schedule. Those metrics make it possible to carry out a more precise analysis concerning the properties of the resource flow network in terms of its robustness to disruptions in comparison with a well known flexibility indicator, *flex*.

Key words: resource-constrained project scheduling, robust scheduling, resource allocation.

1. Introduction

An inherent feature of project execution is its uncertainty connected with the customers' changing requirements, erroneous estimation of task durations, temporary unavailability of resources (e.g., machinery failures) [1] etc. In order to compensate the effect of unpredictable production disruptions, robust scheduling is used. A robust schedule is developed in the phase of project planning, before project execution starts. A robust schedule is defined as such an arrangement of tasks which, owing to its properties, is robust to disruptions in the production process [2]. Robustness is here understood as a schedule ability to compensate the effects of minor increases in the task duration, as such increases may be caused by factors out of control of the production process manager [3]. The authors also develop robustness metrics designed to minimise changes in a schedule caused by fluctuations in task durations.

The paper analyses the Resource-Constrained Project Scheduling Problem (RCPSP). Robust scheduling for an RCPSP problem includes two optimisation phases, designed to develop a schedule which would be as robust to production disruptions as possible:

1) robust resource allocation – appropriate allocation of resources for the execution of individual tasks;
2) robust buffer allocation – arrangement of time and resource buffers; buffer allocation is preceded by resource allocation.

The paper discusses issues connected with the allocation of resources to tasks and the effect of such allocation on schedule robustness. A mathematical model of the resource allocation problem is described, and for that model robustness metrics are presented, both already known and authors-proposed ones.

Resource allocation is important phase of production planning which may cause decreasing of instability cost (storage costs, organizational costs, costs related to agreements with subcontractors etc.). The goal of this article is to present new rules and measures of the robust resource allocation problem. Research works in this area are still in a burn-in phase.

2. Formulation of the problem

A project is a unique set of activities (tasks) executed in order to achieve predefined objectives with use of specified resources (human resources, machinery and materials). An activity (task) is an element of a project, representing a separable entity, for which starting and/or completion times are defined. A task is indivisible: its execution must not be interrupted, which means that nonpreemptive scheduling is used here. Further it is assumed that there is only one way to execute an activity (we have to do with a single-mode RCPSP), that is there is no alternative option of task execution with use of other types of resources – this would be typical of what is known as a multi-mode RCPSP [4].

Projects are described as the AON (Activity On Node) network, that is, formally, as an acyclic simple directed graph $G(V,\ E)$, where V is the set of nodes (vertices) representing activities (tasks), and E is the set of arcs (edges) representing succession relations among tasks. A project is supplemented with two formal tasks 0 and $n{+}1$ with zero durations and zero demand for resources; these tasks represent the source and the sink, respectively, of the graph $G(V,\ E)$.

Between projects tasks, end-beginning precedence relations occur with no delay: the successor may begin just upon the predecessor having ended:

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

where s_i is the starting time of the task i, and d_i is the duration of the task i.

*e-mail: marcin_kli@interia.pl

The execution of an activity requires resources which in the same time define constraints of project execution. Resources are renewable and indivisible (e.g. machinery or human resources). At any given time t the use of resources must not exceed their availability:

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

where a_k is the number of available resources of type k ($k = 1, \ldots, K$), $A(t)$ is the set of tasks performed in the time interval $[t-1, t]$, r_{ik} is the demand of the task i for the resources of type k.

The number of resources is constant (and equal to a_k for any $k = 1, \ldots, K$, where K is the number of resource types), irrespective of the loads in previous periods.

Prior to resource allocation, a nominal schedule is developed, which resolves itself to identifying starting times $s_0, s_1, \ldots, s_{n+1}$ for individual tasks, with succession and resource constraints expressed as formulae (1, 2) factored in. The most common optimisation criterion is the minimisation of entire project duration.

A given schedule can be implemented by various resource flow networks, which may vary among themselves in terms of their robustness to disruptions in the production process [4]. Resource allocation to individual tasks is very important for the robustness of the schedule developed. The resource allocation problem for RCPSP has been discussed in a number of research papers [5–9]. Resource flow networks [5] are used as a tool to describe the resource allocation problem. A resource flow network is a network including the nodes of the original project task network $G(V, E)$ and some additional arcs (whose set is denoted with E_R) connecting every two nodes (tasks) between which resource flows occur described as non-negative integers $f(i, j, k)$ for each resource type k, from the task i just ending to the task j just beginning. The set E_R contains those arcs only which are not in the original network $G(V,\ E)$.

In order to formulate the resource allocation problem, the model described with formulae (1, 2) is supplemented with the following constraints [4, 6, 7]:

– for each resource type from the set of K types, the sum of all resources of the type leaving the formal source equals the sum of resources of the type entering the formal sink and amounts to a_k (total availability of resources of type k):

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

– for each resource type k, the sum of all resources of the type entering a given node representing an actual activity equals the sum of resources of the type leaving that node and amounts to r_{ik} (demand of the task i for resources of type k):

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

3. Robust resource allocation

Even with a single resource type, the robust resource allocation problem for RCPSP scheduling is a strongly *NP*-difficult problem [4]. While allocating resources, it is assumed that the absence of succession relations between two tasks is advantageous, as a delay, if any, in the completion of one task has no effect on the starting time of the other. Each new arc in the set E_R means a new succession constraint, which reduces schedule robustness. Consequently, the problem of optimum resource allocation is reduced to the problem of minimising the number of additional arcs [4, 7]. Robust resource allocation algorithms allocate the same resources to tasks connected by succession relations [8] and maximise sums of flows between individual tasks [4]. The problem of minimising the number of arcs in the task network is also solved with integer programming methods [7].

In order to limit the range of search, unavoidable arcs are identified first. The tasks i and j are connected with an (unavoidable) arc in the resource flow network, if so forced by the schedule for which the resource allocation is performed. Such a situation occurs, when, at the starting time of the task j ($t = s_j$), the number of available resources of a given type k (with the resources which have executed the task i excluded) is less than the demand of the task j for resources of that type. Formally, it can be written as [7]:

$$(i, j) \in E_U \Leftrightarrow \exists k : \left(a_k - \sum_{l \in J_t} r_{lk} - \max\left(0, r_{ik} - \sum_{p \in L_t} r_{pk}\right)\right) < r_{jk} \tag{5}$$
$$\forall i, j \in V : (s_j \geq s_i + d_i), \quad (i, j) \notin E,$$

where E_U is the set of unavoidable arcs ($E_U \subset E_R$), J_t is the set of tasks being executed at the time $t = s_j$, L_t is the set of such tasks p whose starting times fulfil the relation $s_j > s_p \geq s_i + d_i$, that is, at the time $t = s_j$ these tasks may use resources allocated to the task i (as they have started after the resources were released by the task i).

Table 1 sets forth information on the tasks performed in a project example, which will support the analysis of the robust resource allocation problem.

Table 1
Example of a project execute with a single type of resources

Activity i	d_i	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	–

Figure 1 gives an example of the Gantt schedule for the project specified in Table 1. The resource flow network for that schedule is presented in Fig. 2.

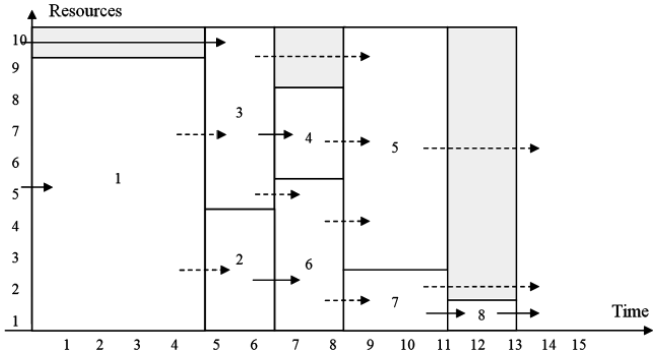

Fig. 1. The Gantt schedule with resource allocation for the project example

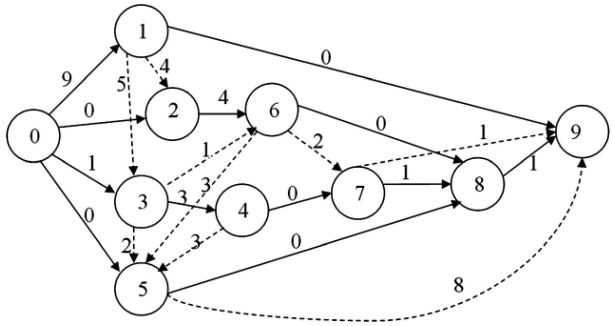

Fig. 2. A resource flow network for the schedule in Fig. 1

In Fig. 1 and Fig. 2, additional arcs (the elements of the set E_R) in the resource flow network are marked with broken lines. For the resource flow network illustrated in Fig. 1, the set E_R comprises 9 arcs: (1, 2), (1, 3), (3, 5), (3, 6), (4, 5), (5, 9), (6, 5), (6, 7) and (7, 9). Among them, unavoidable arcs are: (1, 2), (1, 3), (3, 6), (4, 5), (5, 9), (6, 5) and (7, 9). The arcs (3, 5) and (6, 7) only may theoretically be deleted from the set E_R if the resource allocation is modified.

As part of the research into resource allocation, robustness metrics and algorithms are constructed minimising the number of additional arcs. The authors draw the reader's attention to the fact that certain arcs in the set E_R have a lesser effect on schedule robustness. They are such arcs (i, j) (connecting the task i with task j) for which:

- the task j is in the set of all successors of the task i;
- between the task i and the task j, there do occur resource flows, but these resources are idle for a time.

In the example shown in Fig. 1 and Fig. 2, (3, 5) is such an arc, because in the time interval from 7 to 9, the resources 9 and 10 are idle (the area shaded in grey in Fig. 1).

The resource allocation illustrated in Fig. 1 and Fig. 2 may be improved. For instance, these arcs in the set E_R which are not unavoidable, that is the arcs (3, 5) and (6, 7), theoretically, might be deleted from the set E_R if the resource allocation is modified. The arc (3, 5) only slightly reduces schedule robustness, because a two-unit time buffer occurs along the resource flow. Further, an attempt at eliminating the arc (3, 5) causes a new arc (3, 7) to be created, which means that the cardinality of the set E_R would not decrease. Thus, the minimum cardinality of the set E_R is 8. On the other hand, the deletion of the arc (6, 7) may render the scheduling more robust.

Figure 3 presents the Gantt schedule with the minimum number of additional arcs, with the arc (6, 7) deleted. Figure 4 depicts the resource flow network for that schedule.

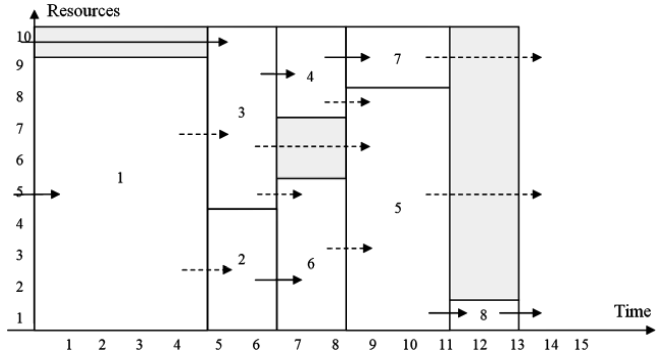

Fig. 3. The Gantt schedule with the minimum number of additional arcs

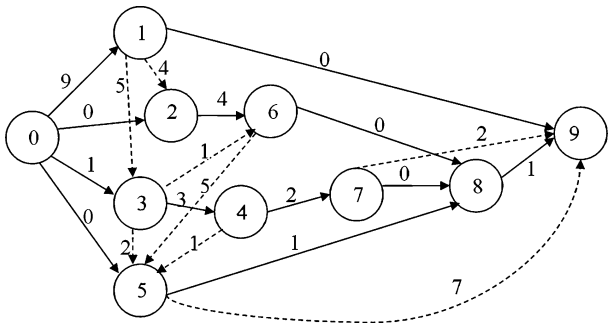

Fig. 4. A resource flow network with the minimum number of additional arcs

Resources from a given node should first be transferred to its direct successor. Absence of resource flow between activities directly connected with the succession relation may prove a solution adverse to the robustness of resource allocation. In practice, succession relations are defined by technological considerations.

The quality of resource allocation is assessed with use of robustness metrics, which should take into consideration certain principles of robust resource allocation, including:

- the same resources should be used to execute tasks which are directly connected with the succession relation;
- flows between individual activities should be maximised, with unavoidable arcs included;
- the number of additional arcs should be minimised;
- at any given time, those resources should be allocated first which have not been used at an earlier time.

4. Resource allocation robustness metrics

While assessing schedule robustness, the effect of possible disruptions on the robustness is determined. Resource

allocation-specific metrics are used most commonly, as they take into consideration a resource flow network. It is also possible to use indicators applicable in reactive scheduling. If this is the case, disruptions in the production process (e.g., changes in task durations) are generated randomly and scheduling stability indicators are computed.

Flex (flexibility) is used to measure resource allocation robustness [8]. The value of flex is computed as the quotient of the number of such task pairs in the resource flow network between which there is no succession relation, to the total number of task pairs in the project. The flex indicates what proportion of tasks has no succession relations to the other tasks. The higher flex, the less the interdependence among tasks, and the higher the schedule robustness. The flex maximisation problem is actually the problem of minimising the number of additional arcs. For the RCPSP problem considered, where a topological task arrangement and no-delay end-beginning succession relations occur, flex may be computed as follows:

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

where $\frac{n \cdot (n-1)}{2}$ is the number of all possible task pairs in the project, $\#(E \cup E_R)$ is the number of arcs in the resource flows network (including additional arcs).

Flexibility is a simple metric. It often happens that for a given nominal schedule, there exist numerous resource allocations with the same flex, but varying robustness. The flex ratio does not take into consideration, e.g., the effect of individual additional arcs on the robustness.

A more advanced approach consists in using the objective function of reactive scheduling as resource allocation robustness metric. Among the objectives of reactive scheduling is schedule stability, which is as precise as possible execution of the planned schedule. Under this approach, the effect of lengthening the durations of individual activities on production stability is determined [7]. Instability cost is determined by experiments for various production disruption scenarios generated, for instance, based on available statistical data. The use of such objective function determined by simulation is time-consuming and depends on simulation parameters (disruptions), whose generation prior to the implementation of a schedule is loaded with uncertainty. Not always is the statistical data concerning production disruptions available. Accordingly, it is reasonable to develop resource allocation metrics determined statically and taking into consideration the effect of lengthening the durations of individual activities on the production stability. In the construction thereof, the authors assume that planned durations of all activities are subject to lengthening with the same probability [9].

As the robustness metric, the stab_1 indicator is used, computed as the aggregate of the delays in the commencement of all tasks relative to the planned commencement times, assuming that each duration of a task is lengthened by one unit of time [9]:

$$stab_1 = \sum_{i=1}^{n+1} (s_i^{all} - s_i), \tag{7}$$

where s_i^{all} is the actual commencement time of the task i with all task durations lengthened by 1.

For the purposes of determining the value of $stab_1$, a modified schedule is created with a resource flow network, taking into consideration the modified task durations (the duration equals $d_j + 1$ for each task j, for $j = 1 \ldots n$). A simpler variant of the metric is the duration of the entire project under the modified schedule:

$$stab_2 = s_{n+1}^{all}. \tag{8}$$

Another proposed robustness metric is the function $stab_3$ computed as the number of the tasks delayed (shifted) as a result of a delay of each activity by 1 unit of time:

$$stab_3 = \sum_{j=1}^{n} \left(\sum_{i=1}^{n+1} (s_i^j - s_i) \right), \tag{9}$$

where s_i^j is the commencement time of the task i assuming lengthening of duration of the task j by 1.

The number of delayed (shifted by 1 unit of time) tasks when a given task duration is lengthened by 1 is determined on the assumption that the other tasks are executed as planned. Time shifts are determined based on the analysis of the resource flow network.

A schedule whose resource allocation minimises $stab_1$, $stab_2$ or $stab_3$ is robust to disruptions. With such allocation, minor duration lengthening, if any, of some tasks has a lesser effect on the stability of the schedule being implemented (on delays of other tasks).

5. Conclusions

Resource allocation is that phase of optimisation process in which a robust allocation of available resources to planned project tasks is searched for. Any identified robust resources allocation may materially support a robust allocation of buffers, which ensures timely execution of individual tasks and the entire project.

In the paper, the authors analyse the resource allocation problem for an exemplifying project, indicating at allocation principles which may enhance the robustness of the resource flow network obtained. Then, criteria (metrics) for assessing a resource flow network are defined; the authors deem these criteria more useful than the metrics used in the research to date. The proposed robustness metrics are designed to guarantee that the effect of the duration lengthening for individual tasks on the stability of the final schedule (delays in other tasks) is minimised.

REFERENCES

[1] D. Gąsior and J.Józefczyk, "Application of uncertain variables to production planning in a class of manufacturing systems", *Bull Pol. Ac.: Tech.* 57 (3), 257–263 (2009).

[2] M. Jensen., "Improving robustness and flexibility of tardiness and total flow-time job shops using robustness measures", *Applied Soft Computing* 1, 35–52 (2001).

[3] M. Al-Fawzan and M. Haouari, "A bi-objective problem for robust resource-constrained project scheduling", *Int. J. Production Economics* 96, 175–187 (2005).

[4] R. Leus and W. Herroelen, "Stability and resource allocation in project planning", *IIE Transactions* 36 (7), 667–682 (2004).

[5] C. Artigues, P. Michelon, and S. Reusser, "Insertion techniques for static and dynamic resource-constrained project scheduling", *Eur. J. Operational Research* 149 (2), 249–267 (2003).

[6] K. Braeckmans, E. Demeulemeester, W. Herroelen, and R. Leus, "Proactive resource allocation heuristics for robust project scheduling", *Research Report KBI_*0567, K.U.Leuven, Leuven, 2005.

[7] F. Deblaere, E. Demeulemeester, W. Herroelen, and S. Van De Vonder, "Proactive resource allocation heuristics for robust project scheduling", *Research Report KBI_*0608, K.U.Leuven, Leuven, 2006.

[8] N. Policella, "Scheduling with uncertainty – a proactive approach using partial order schedules", *Phd Thesis*, University La Sapienza, Rome, 2005.

[9] M. Klimek and P. Łebkowski, "Robust resource allocation algorithms for the resource-constrained project scheduling problem," *Automatics, Semiannual of the AGH University of Science and Technology of Kraków* 13 (2), 371–379 (2009), (in Polish).