A Serial Schedule Generation Scheme for Project Scheduling in Disaster Management

Niels-Fabian Baur and Julia Rieck

University of Hildesheim, Germany
`{baur,rieck}@bwl.uni-hildesheim.de`

Keywords: disaster management, schedule generation scheme, flexible resource profiles.

1 Project Scheduling in Disaster Management

Due to climatic changes and a concomitant accumulation of extreme weather events, natural disasters, e.g., hurricanes and floods are a growing threat worldwide. According to the survey of Altay and Green, disasters can be described as large-scale events that pose an unusually high threat to life and health as well as to material assets. A particular challenge is the uncertainty of the events as well as the difficulty to predict a disasters impact. Four phases can be identified in the lifecycle of disaster management (i.e., mitigation, preparedness, response, and recovery). In particular, the response phase (post-disaster), where activities must be coordinated and information exchanged quickly, is considered in the literature, e.g., in the fields of infrastructure protection and medical care (Altay and Green 2006). Here, models and decision support systems can be used to directly reduce the impact of disasters. Therefore, the response phase is addressed in the following.

Using governmental emergency plans, necessary activities can be pre-defined that have to be carried out immediately after a disaster. For successful planning, it makes sense to visualize their precedence constraints by a project with a corresponding network. However, the execution of the activities requires suitable resources, the emergency forces. When responding to a disaster, it is helpful to have as many workforces as possible to carry out the necessary relief measures. Volunteers can constitute important resources and therefore be an effective complement to the professional forces in disaster relief. Hence, a successful response should integrate voluntary helpers. They must be assigned to activities and start times of activities must be determined. Consequently, a combined workforce and project scheduling problem arises. Due to the high complexity of the resulting problem, we have developed a serial schedule generation scheme (SGS) that finds feasible solutions even for large problem instances in reasonable time.

2 Problem Definition and Solution Approach

As described in Section 1, we consider a combined workforce and project scheduling problem (cf. Baur and Rieck 2019 for the mathematical model). It is assumed that projects with n real activities $i, j = \{0, \dots, n+1\}$ and a set of reasonable precedence constraints E can be predefined. All real activities (e.g., fill sandbags and carry sandbags) require volunteers $k \in K$ to be carried out. Whether a volunteer can be assigned to an activity depends on two important aspects. Since voluntary helpers constitute partial renewable resources, each one has a defined time interval in which it is available. A resource k can be assigned to an activity at time t if $\theta_{kt} = 1$ applies. The other precondition for an assignment is that the resource is suitable for an activity. Every activity has a corresponding set of skills S_i that are needed to process it. Exemplary skills are physical fitness and driving licenses. A volunteer has to declare an associated level L_{ks} for all predefined skills $S \supseteq S_i$, which indicates to what extent the skill is mastered. According to Mansfield, the levels range from "not demonstrated" (i.e., $L_{ks} = 0$) to "outstanding" (i.e., $L_{ks} = 2$). A volunteer with a low skill level needs more time for the same workload (Mansfield 1996). Consequently, only if at least one volunteer with required skills is available, an activity can be processed. It is completed when the estimated total workload D_i is reached for every required skill.

Even if all predecessors $Pred(i)$ of activity i have been completed, a delay of the start S_i may be necessary, if there is not at least one suitable volunteer available. If an activity i already started but is not yet finished and no resource can be assigned at any time, the activity must be interrupted. Figure 1 shows exemplary interruptions of activity $i = 1$. It starts when the first resource is assigned at $t = 1$. At time $t = 2$ and $t = 4$ interruptions occur, as no suitable worker can be selected. After six time periods (i.e., $P_1 = 6$), the total processing time of $D_1 = 4$ is covered and the activity is completed.

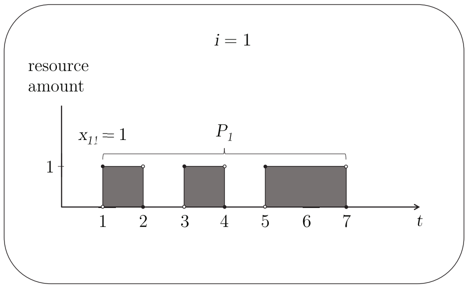

Fig. 1. Interruptions of an activity

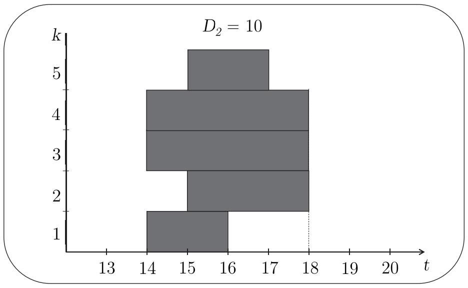

Fig. 2. Multiple assignments to an activity

If more than one resource is available at a given time, a team of several volunteers with different skill levels can be assigned to an activity. This would lead to a reduction of the expected activity duration P_i, which is variable and no deterministic parameter. The size of the assigned team is not constant during the processing time of an activity i what can be seen in Figure 2. It visualizes another example, where five resources are assigned simultaneously during the execution of activity $i = 2$. Although the estimated total processing time is $D_2 = 10$, the activity can be completed within four time units (i.e., $P_2 = 4$). Note that the number of resources assigned to $i = 2$ differs over time. While in period 14 only three volunteers are assigned, in period 15 there are five resources working in total. Therefore, the problem under consideration is a problem with flexible resource profiles (cf. (Naber and Kolisch 2014)). Besides the considered skills and the possible interruptions of an activity, the multiple resource assignments and variable activity duration are the most important characteristics of the problem. These properties make the problem more realistic, but also more difficult to solve. For this purpose, we implemented the SGS shown in Algorithm 1 to create feasible solutions even for large instances in decent time.

In the initialization step, the fictitious project start $i = 0$ is scheduled and added to the set of already completed activities $\mathcal{C}$. The schedule of all completed activities ST contains the corresponding start time $S_0 = 0$. Furthermore, the predecessors $Pred(i)$ of all nodes $i \in V$ are determined. The main step from line 3 is executed until all activities have been completed. At the beginning of his step, the eligible set $\mathcal{E}$ of activities is determined from which all predecessors have already been completed. The earliest start times ES_j of all activities $j \in \mathcal{E}$ are calculated in line 5. The activity with the highest priority is selected for the further procedure. The priority rule of the earliest start time (EST) was applied for first computational studies. For the selected activity j, the set σ of all skills for which the required working time D_j has not yet been reached is defined. T_j describes the set of time periods t at which j is actively processed by one or more resources and D_j^s represents the number of working hours remaining for each skill.

Algorithm 1 Serial Schedule Generation Scheme

```
1:  set $\mathcal{C} := \{0\}$, $ST^{\mathcal{C}} := (0)$;
2:  determine set of all predecessors $Pred(i)$ for activities $i \in V \setminus \{0\}$;
3:  while $\mathcal{C} \neq V$ do
4:      determine $\mathcal{E} := \{i \in \bar{\mathcal{C}} | Pred(i) \subseteq \mathcal{C}\}$;
5:      calculate earliest start $ES_j := \max_{i \in Pred(i)}(ST_i + P_i)$ of all $j \in \mathcal{E}$;
6:      choose $j \in \mathcal{E}$ with highest priority;
7:      determine $\sigma := \{s \in S_j\}$, $T_j := \emptyset$ and $D_j^s = D_j$ for all $s \in \sigma$;
8:      while $\sigma \neq \emptyset$ do
9:          for $t = ES_j$ to $\bar{d}$ do
10:             for $k \in K$ with $\theta_{kt} = 1 \wedge \sum_{s \in \sigma} L_{ks} > 0$ do
11:                 set $r_{jkt} := 1$, $\theta_{kt} := 0$ and $T_j := T_j \cup \{t\}$;
12:                 for $s \in \sigma$ with $L_{ks} > 0$ do
13:                     calculate $D_j^s := D_j^s - L_{ks}$;
14:                     if $D_j^s \leq 0$ then
15:                         set $\sigma := \sigma \setminus \{s\}$ and $P_j := t + 1 - ST_j$;
16:     set $\mathcal{C} := \mathcal{C} \cup \{j\}$ and $ST_j := \min_{t \in T_j} t$;
    return $ST^{\mathcal{C}}$.
```

The inner loop starting from line 8 is executed until σ is an empty set, thus the required total working hours for each skill have been met ($D_j^s \leq 0,\ \forall s \in \sigma$). From the earliest start until an activity can be completed or the maximum planning horizon is reached, all available resources k that have at least one of the required skills ($\sum_{s \in \sigma} L_{ks} > 0$) are considered one after the other. If no resource is available, the procedure continues with the next period. In line 11, the first resource found is assigned to the activity j at the current time t ($r_{ikt} = 1$ applies). Consequently, the volunteer is no longer available for other activities in this period (i.e., $\theta_{kt} := 0$). The lines 12-15 update the number of working hours still needed for each skill that is mastered by resource k and required for activity j. Once the total working time for a skill has been reached, the skill is removed from set σ and no longer needs to be considered. If σ is an empty set, activity j can be terminated and added to the set $\mathcal{C}$ in line 16. The start time of the activity j is defined as the time of the first resource assignment to j. When all activities are completed, the procedure terminates and returns the schedule of all start times.

3 Computational Results

For our computational study, we created 20 instances with $n = \{30, 60\}$ real activities on the basis of the PSPLIB benchmark (Kolisch and Sprecher 1996). The instances are supplemented by problem-specific parameters. For example, the number of considered skills is randomly set from 3 to 5. Under the assumption that the default skill level $L_{ks} = 1$ is the most common in reality, it gets the highest generation probability. The levels $L_{ks} = 0$ and $L_{ks} = 2$ are the least likely. The availability of resources is randomly determined within $\{8, 9, \ldots, 18\}$ time units without breaks. The SGS was implemented in C++ with Visual Studio 2019. The comparison results were generated with CPLEX 12.9 in GAMS 25.1 within a time limit of 7200 seconds. The tests were carried out on a server (two 2.1 GHz processors and 384 GB of RAM) using up to 16 threads.

Table 1 shows the obtained results. Instances with numbers 1 to 10 include 30 real activities. Instances 11 to 20 include 60 real activities. The SGS found feasible solutions for all instances within 100 s, which can be seen in the column “CPU”, whereas CPLEX has only found a solution for five instances with 30 activities and no solution for the larger instances. The objective function values of the procedures can be taken from the columns $F(x)$. The objective is to minimize the project duration and thus to cope with the disaster as soon as possible. The column “Gap” shows the deterioration of the solution found by the SGS compared to the solution of CPLEX. For instances 4 and 5, the SGS found an equally good solution after 24 respectively 60 s, as CPLEX did after 7200 s. Only for instance 7, the SGS found a clearly worse (61.5%) solution than CPLEX.

Table 1. Comparison of SGS and CPLEX solutions for instances with $n = 30$ and $n = 60$

	SGS		**CPLEX**		
no.	$F(x)$	CPU [s]	$F(x)$	CPU [s]	Gap [%]
1	46	42	–	7229	–
2	29	27	–	7212	–
3	33	36	–	7223	–
4	21	24	21	2890	0.0
5	38	60	38	7230	0.0
6	22	4	20	2228	10.0
7	42	7	26	7215	61.5
8	29	7	–	7228	–
9	27	26	26	3519	3.8
10	32	5	–	7247	–
11	50	30	–	7883	–
12	43	72	–	7807	–
13	51	25	–	8065	–
14	52	70	–	7876	–
15	78	22	–	7954	–
16	37	15	–	8168	–
17	81	27	–	8121	–
18	48	16	–	8097	–
19	100	76	–	8009	–
20	59	24	–	8089	–

4 Conclusion and Outlook

The abstract introduces a serial schedule generation scheme for a particular problem with skills, skill levels, possible interruptions of activities, and variable activity durations. The results of the procedure were compared to the results of CPLEX. The next step is the development of a metaheuristic, which is able to improve the found solution in reasonable time. In addition, the problem should be adapted to the dynamic and stochastic characteristics of a disaster by transforming the currently static and deterministic model into a dynamic formulation with stochastic components.

References

Altay, N., Green, W.,G., 2006, “OR/MS research in disaster operations management”, *European Journal of Operational Research*, Vol. 175, pp. 475-493.

Baur, N.-F., Rieck, J., 2019, “Project Management with Scarce Resources in Disaster Response”, submitted to *Operations Research Proceedings 2019*.

Kolisch, R., Sprecher, A., 1996, “PSPLIB – A project scheduling problem library”, *European Journal of Operational Research*, Vol. 96, pp. 205-216.

Mansfield, R.S., 1996, “Building competency models: Approaches for HR professionals”, *Human Resource Management*, Vol. 35, pp. 7-18.

Naber, A., Kolisch, R., 2014, “MIP models for resource-constrained project scheduling with flexible resource profiles”, *European Journal of Operational Research*, Vol. 239(2), pp. 335-348.