Solving the Multi-mode Resource Investment Problem with Constraint Programming

Patrick Gerhards[1]

Helmut Schmidt University Hamburg, Germany
patrick.gerhards@hsu-hh.de

Keywords: Project Scheduling, Multi-mode Resource Investment Problem, Constraint Programming

1 Introduction

When dealing with project scheduling problems, often a fixed deadline for the project completion time is imposed. The main concern of the project manager is to plan the use of resources in such a way, that the project finishes on time and the project costs are minimised. In this work, we will investigate how we can solve a problem of this kind - the multi-mode resource investment problem (MRIP) - using mixed-integer programming (MIP) and constraint programming (CP) techniques. Since the efficiency of constraint programming solvers increased significantly in recent years, we want to study if they are a suitable procedure when solving this problem type.

2 Multi-mode Resource Investment Problem

The MRIP is a project scheduling problem and an extension of the resource investment problem (RIP). The RIP, also known as resource availability cost problem (RACP), was introduced by Möhring (1984) and has many practical application cases (e.g. software development or construction projects). For the multi-mode variant, Hsu and Kim (2005) presented a priority rule heuristic and Qi et al. (2015) applied a modified version of the particle swarm optimisation metaheuristic to the MRIP. Kreter et al. (2018) tested MIP as well as CP formulations of the RIP (and some of its extensions) and showed that CP techniques work especially well. They solved many of the open instances to optimality.

Next, we give a formal definition of the MRIP. An instance of the MRIP consists of a set of activities $A = \{0, \ldots, n+1\}$ with precedence relations $E \subset A \times A$ among them. The activities are nonpreemptable and for each activity i, there is a set of modes M_i. Depending on the chosen mode $m \in M_i$, the activity processing duration d_{im} can vary. There are two types of resources that are required by the activities in the MRIP: the renewable resources in $\mathcal{R}$ replenish after each period and are useful to model workers or machines. Non-renewables resources $\mathcal{R}^n$ are consumed by activity execution and do not replenish. The amount of required resource units of activity i depends on the mode m and the resource $k \in \mathcal{R}$ ($k \in \mathcal{R}^n$) and is denoted by r_{imk} (r^n_{imk}). For the renewable resources, the peak resource consumption in all periods needs to be lower than or equal to the resource capacity a_k allocated to the project. The renewable resource costs are computed by multiplying the capacity with the resource unit cost factor c_k. Similar, the non-renewable costs are also the product of the resource unit cost factor c^n_k and the total non-renewable resource consumption a^n_k. In the MRIP, there is a also a deadline D given that restricts the project completion. Using forward and backward calculation (Kelley 1963), we can compute bounds on the earliest start (EST_i) and latest finish (LFT_i) times of the activities. The goal is to find a precedence feasible schedule and a mode assignment that minimises the resource costs.

We present a mathematical model for the MRIP using so-called *pulse* variables x_{imt}. It is an adaption of a model for the resource constrained project scheduling problem (Artigues 2017). In preliminary experiments, the model based on *pulse* variables achieved better results than the models with *on-off* or *step* variables. For each activity $i \in A$, mode $m \in M_i$ and period $t \in [EST_i, LFT_i - d_{im}]$ we introduce the binary decision variable x_{imt} that is equal to 1 if and only if activity i is processed in mode m and starts in period t.

Let us now show a model that can be used as a mixed-integer program to solve the MRIP.

$$\min \sum_{k \in \mathcal{R}} c_k \cdot a_k + \sum_{k \in \mathcal{R}^n} c_k^n \cdot a_k^n \tag{1}$$

$$s.t. \sum_{m \in M_i} \sum_{t=ES_i}^{LF_i - d_{im}} x_{imt} = 1 \quad \forall i \in A \tag{2}$$

$$\sum_{m \in M_i} \sum_{t=ES_i}^{LF_i - d_{im}} x_{imt}(t + d_{im}) \leq \sum_{m \in M_j} \sum_{t=ES_j}^{LF_j - d_{jm}} x_{jmt} \cdot t \qquad \forall (i,j) \in E \tag{3}$$

$$\sum_{i \in A} \sum_{m \in M_i} \sum_{t=ES_i}^{LF_i - d_{im}} x_{imt} \cdot r_{imk}^n \leq a_k^n \quad \forall k \in \mathcal{R}^n \tag{4}$$

$$\sum_{i \in A} \sum_{m \in M_i} \sum_{q=\max(ES_i, t-d_{im}+1)}^{\min(t, LF_i - d_{im})} x_{imq} \cdot r_{imk} \leq a_k \quad \forall k \in \mathcal{R}, t = 0, \ldots, D \tag{5}$$

$$a_k \geq 0, \quad a_k^n \geq 0 \quad \forall k \in \mathcal{R}^n \tag{6}$$

$$x_{imt} \in \{0,1\} \quad \forall i \in A, \forall m \in M_i, t = ES_i, \ldots, LS_i \tag{7}$$

In the objective function (1), we minimise the sum of the resource costs. The renewable part is the product of the peak resource usage a_k and the given resource cost factor c_k. Similarly, we multiply the amount of consumed non-renewable resource units a_k^n with its resource cost factor c_k^n to get the non-renewable resource costs. The constraints (2) ensure that each activity is executed in exactly one mode and exactly one start time is chosen. We model the precedence constraints in a aggregated way and they are displayed in (3): if $(i,j) \in E$, then the finish period of activity i (left side of the inequality) has to be lower than or equal to the start period of activity j (right side). In constraint (4), we compute the non-renewable resource consumption for each non-renewable resource and (5) shows the renewable resource consumption in each time period $t \in [0, D]$. Lastly, (6) - (7) depict the decision variables.

3 Constraint Programming Model

Next, we present a constraint programming model. We use the software IBM ILOG CPLEX CP Optimizer (cf. Laborie et al. (2018)) to model and solve the MRIP using CP-based techniques. The modelling language of CPLEX CP Optimizer offers the use of so-called *interval variables*. They can be used to model the start and finish time of an activity and with the keyword *size*, it is possible to specify the length of the interval (i.e., the difference between the finish and start time). With *optional*, we can declare that an activity can be left unperformed (useful in the context of different modes) and `presenceOf` shows, if an interval variable is performed or not. Let us introduce the decision variables used in our model: With $act[i]$ we define a interval variable for each activity $i \in A$

(see (14)). Since activities can be performed in multiple modes, we introduce in (15) an optional interval variable $mode[i,m]$ for each mode with a specific duration d_{im}. Among interval variables, we can use time expressions such as `endBeforeStart` to model the precedence restrictions as seen in (10). With the `alternative` expression, we can link the *act* and *mode* interval variables. The constraint $\texttt{alternative}(b, \{b_1, \ldots, b_n\})$ ensures that if interval variable b is present, then exactly one of the interval variables in $\{b_1, \ldots, b_n\}$ is also present and their start and end times coincide. This expression is used in (9) to ensure that we choose exactly one processing mode for each activity. To model the peak resource consumption, we use real valued decision variables a_k and a_k^n (see (13)). We make use of a `cumulative` function named $renewUsage_k$ in (12) to represent the renewable resources. There, we sum up over the resource consumptions of the present interval variables with the so called $\texttt{pulse}(a,h)$ expression (that adds the amount h between the start and end time of interval variable a). The non-renewable resources are modelled by summing up the respective resource consumptions of all present mode interval variables in (11). Finally, in the objective function (8), we sum up the resource costs for the peak resource usage of the renewable and non-renewable resources.

$$\min \sum_{k \in \mathcal{R}} c_k \cdot a_k + \sum_{k \in \mathcal{R}^n} c_k^n \cdot a_k^n \tag{8}$$

$$s.t. \quad \texttt{alternative}(act[i], \{mode[i,m] : m \in M_i\}) \qquad \forall i \in A \quad (9)$$

$$\texttt{endBeforeStart}(act[i], act[j]) \qquad \forall (i,j) \in E \quad (10)$$

$$\sum_{i \in A} \sum_{m \in M_i} \texttt{presenceOf}(mode[i,m]) \cdot r_{imk} \leq a_k^n \qquad \forall k \in \mathcal{R}^n \quad (11)$$

$$renewUsage_k = \sum_{i \in A} \sum_{m \in M_i} \texttt{pulse}(mode[i,m], r_{imk}) \leq a_k \qquad \forall k \in \mathcal{R} \quad (12)$$

$$a_k \geq 0 \quad \forall k \in \mathcal{R} \quad a_k^n \geq 0 \qquad \forall k \in \mathcal{R}^n \quad (13)$$

$$\texttt{interval } act[i] \qquad \forall i \in A \quad (14)$$

$$\texttt{interval } mode[i,m] \textit{ optional size } d_{im} \qquad \forall i \in A, \forall m \in M_i \quad (15)$$

The CPLEX CP Optimizer software features an automatic search that is complete and tunes its parameters automatically. It uses propagation of the temporal network, filtering algorithms for the cumulative resource constraints and large neighborhood search techniques to solve complex scheduling problems (Laborie et al. 2018).

4 Computational Experiments

In order to test the performance of the two approaches presented above, we used the benchmark instances of the RIPLib dataset[1]. It features MRIP instances with 30, 50 and 100 activities, 3 or 6 modes per activity and up to 8 renewable resources. In total, 4 950 instances were used in our experiments. We used version 12.9.0 of the CPLEX CP Optimizer solver to solve the CP model depicted above and Gurobi 9.0.0 to solve the MIP model presented before. The solvers were executed on an Intel Xeon Silver 4214 CPU running at 2.20 GHz and the thread count for each solver was restricted to 1. As a stopping criterion, we used time limits of 60, 600 and 3 600 seconds. In Table 1 we depict the portion of instances that were solved to optimality by the MIP or CP solver. Surprisingly, the CP method solved almost twice as many instances in 60 seconds than the MIP solver in 3 600 seconds. With the 1 hour time limit, CP solved almost one third of the instances

[1] https://riplib.hsu-hh.de/

Table 1. Percentage of instances solved to optimalitiy

	Max runtime in seconds		
Method	60	600	3 600
MIP	0.9 %	4.3 %	9.3 %
CP	14.6 %	22.6 %	28.4 %

to optimality. It shows clearly, that CP outperforms the MIP approach on these instances. Further results will be presented at the conference due to space limitations and can also be found in (Gerhards 2020).

References

Artigues, C., 2017, "On the strength of time-indexed formulations for the resource-constrained project scheduling problem", *Operations Research Letters* , Vol. 45, No. 2, pp. 154-159.

Gerhards, P., 2020, "The multi-mode resource investment problem: a benchmark library and a computational study of lower and upper bounds", *OR Spectrum*, Vol. 42, No. 4, pp. 901-903.

Hsu, C. C., D. S. Kim, 2005, "A new heuristic for the multi-mode resource investment problem", *Journal of the Operational Research Society*, Vol. 56 No. 4, pp. 406-413.

Kelley, J. E., 1963, "The critical-path method: Resources planning and scheduling", *Industrial Scheduling*, Vol. 13, no. 1, pp. 347-365.

Kreter, S., Schutt, A., Stuckey, P. J., Zimmermann, J., 2018, "Mixed-integer linear programming and constraint programming formulations for solving resource availability cost problems", *European Journal of Operational Research*, Vol. 266, No. 2, pp. 472-486.

Laborie, P., J. Rogerie, P. Shaw, P. Vilím, 2018, "IBM ILOG CP optimizer for scheduling", *Constraints*, Vol. 23, No. 2, pp. 210-250.

Möhring, R. H., 1984, "Minimizing costs of resource requirements in project networks subject to a fixed completion time", *Operations Research*, Vol. 32, No. 1, pp. 89-120.

Qi, J. J., Y. J. Liu, P. Jiang, B. Guo, 2015, "Schedule generation scheme for solving multi-mode resource availability cost problem by modified particle swarm optimization", *Journal of Scheduling*, Vol. 18, No. 3 pp. 285-298.