Boosting cumulative propagations with cumulative strengthening

Nicolas Bonifas[1,2], Réda Bousseta[3]

[1] LIX, Ecole Polytechnique, 91128 Palaiseau Cedex, France
nicolas.bonifas@polytechnique.edu

[2] IBM France Lab, 9 rue de Verdun 94250 Gentilly, France

[3] EPFL, Lausanne, Switzerland
redabousseta@hotmail.com

1 Introduction

The cumulative resource is at the heart of constraint-based scheduling. Given n tasks, a cumulative resource of *capacity* C where tasks have respective *demands* c_i is said to be satisfied if a schedule exists such that at all time points, the sum of the demands of the tasks under execution does not exceed C.

Having fast propagators with a strong pruning power is essential in building effective constraint programming systems. A number of works in recent years [1, 2, 3] made use of redundant cumulative resources, that is changing the demands of certain tasks on the cumulative resource while ensuring that no feasible solution is lost, to compute stronger bounds or tighten the instance to get stronger propagations. This technique, sometimes called "cumulative strengthening", is especially useful when used with propagations which rely on computing energetic bounds, such as edge finding or energy reasoning.

Two questions arise when using this technique to improve temporal bounds: 1) how should one generate the reformulations, which of the techniques in the literature will generate strong enough reformulations at a reasonable computing cost? 2) which of these reformulations will be effective for a particular instance?

In order to get some insight on these questions, we decided to experiment and characterize the maximum propagation power that we can obtain from a reformulation-based approach.

Our approaches are tailored to RCPSP instances, that is scheduling problems consisting of n tasks of respective lengths p_i, with R cumulative constraints of respective capacity C_i, the demand of the ith task on the jth resource being denoted $c_{i,j}$, and a set of precedences $Prec$, with $(a, b) \in Prec$ meaning that task a must complete before task b can start.

This note presents preliminary results on boosting the propagation power of common propagations with this technique.

2 Generating and selecting redundant cumulative resources

We start from a very simple model to compute a global bound for a RCPSP instance: the only constraint family in our model represents "tasks configurations", that is set of tasks which can be executed simultaneously. Variables are shown in bold letters.

$$\begin{array}{lll} \max & & \displaystyle\sum_{i=1}^{n} p_i \mathbf{h}_i \\ \text{s.t.} & \forall P \in \mathcal{P} & \displaystyle\sum_{i=1}^{n} P_i \mathbf{h}_i \leq 1 \end{array}$$

In practice, the set of configurations $\mathcal{P}$ is generated by row generation: we start with an empty set and solve the above LP. Each time we find a solution, we try solving the following

ILP with the h_i coefficient obtained as a result of the LP. If the ILP is feasible, we had its solution P to the set $\mathcal{P}$, otherwise we stop generating configurations and return the bound and h_i values.

In order to generate powerful reformulations, we take as much information as possible into consideration: this includes information about all the cumulative resources in the RCPSP instance, and about the precedence relationships.

The row generation ILP is shown below:

$$\begin{array}{rl}
\text{s.t.} & \sum_{P \in \mathcal{P}} \mathbf{P}_i h_i \\
\forall r \in [1, R] & \sum_{P \in \mathcal{P}} \mathbf{P}_i c_{i,r} \leq C_r \\
\forall (a,b) \in Prec & \mathbf{P}_a + \mathbf{P}_b \leq 1 \\
 & \mathbf{P} \in \{0,1\}^n
\end{array}$$

This is different from the approach in [3] in that there is a new demand per task, instead of aggregating all tasks with the same original demand level.

3 Propagation strengthening: the example of edge-finding

A fruitful approach in using the cumulative strenghtening technique shown above is to tailor the reformulation to the needs of a constraint propagator. An example is shown here with the edge finding propagation.

In its simplest form, the edge finding propagation does the following: given the current earliest start times est_i and latest end times let_i of all tasks, for a set of activity indices $\Omega \subseteq [1, n]$ and $i \notin \Omega$, if

$$\sum_{j \in \Omega \cup \{i\}} p_j c_j > C((\max_{j \in \Omega} let_j) - (\min_{j \in \Omega \cup \{i\}} est_j)),$$

then all activities in Ω end before the end of task i.

In order to boost edge finding using cumulative strengthening, we can reuse the formulation from the previous section and replace the objective function with the goal of maximizing $\sum_{j \in \Omega \cup \{i\}} p_j \mathbf{h_j}$. This result in a different reformulation for each Ω and i.

In practice, this technique is quite costly, but the heaviest work (computing the configurations) can be done only once and only the LP has to be resolved with a different objective. Various techniques enable us to optimize this algorithm further.

We tried this technique on several propagation algorithms and showed that these reformulations can significantly improve their propagation power. For instance, the basic edge-finder with the optimal reformulation can sometimes find propagations that the timetable edge-finder missed!

This surprising result shows the potential of the propagation boosting technique. We obtained similarly interesting results with the precedence energy constraint.

References

[1] Brucker, P. and Knust, S. *A linear programming and constraint propagation-based lower bound for the RCPSP.* European Journal of Operational Research, 127(2):355–362. Elsevier, 2000.

[2] Carlier, J. and Néron, E. *Computing redundant resources for the resource constrained project scheduling problem.* European Journal of Operational Research, 176(3):1452–1463. Elsevier, 2007.

[3] Bonifas, N. and Baptiste, P. *Nouvelles bornes pour le RCPSP par reformulation de ressources cumulatives.* ROADEF, 2014