Resource capacity and time lag propagation techniques for RCPSP problem*

Dmitry Arkhipov[1,2], Olga Battaïa[1], and Ilia Tarasov[1,2]

[1] ISAE-SUPAERO, Université de Toulouse, France
[2] V.A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, Moscow, Russia

Abstract. Constraint programming is one of the most popular approach for solving the well-known Resource-Constrained Project Scheduling Problem (RCPSP). However, since RCPSP is NP-complete, constraint programming is not able to find suitable solutions in reasonable time for large-scale instances. In this paper, we propose some new propagation techniques based on resource capacity and time lags propagation to make constraint programming more efficient for solving RCPSP. We show that these new algorithms are able to make tasks domains tighter or to improve the performance of existing propagators.

Keywords: Project scheduling · Constraint programming · Resource propagation · Domain propagation · RCPSP.

1 Introduction

The Resource-Constrained Project Scheduling Problem is a classic scheduling problem, proved to be NP-hard in [1]. We consider the following satisfiability statement of the problem. There is a set of resources R, the capacity of each resource $r \in R$ is defined by c_r. The set of tasks N have to be processed subject to satisfaction of resource and precedence constraints in time horizon H. For each task $j \in N$, the following parameters are given:

- est_j – the earliest (possible) starting time;
- lct_j – the latest (possible) completion time;
- p_j – processing time;
- a_{jr} – required amount of resource $r \in R$.

For the set of tasks N, *schedule* π is defined if start time $S_j(\pi)$ is assigned to each task $j \in N$.

Precedence relations with time lags for ordered pair of tasks $(i, j) \in N^2$ are defined by real numbers e_{ij}, it means that under any feasible schedule π holds $S_i(\pi) + e_{ij} \leq S_j(\pi)$.

For some pairs of tasks $(i, j) \in N^2$, disjunctive relations g_{ij} are defined. This means that if $g_{ij} = 1$, then tasks i and j cannot be processed simultaneously.

The objective is to find a schedule which satisfies the following constraints:

* Supported by the Russian Foundation for Basic Research (grant 18-37-00295 mol_a)

1. Domain constraint, i.e. for any $j \in N$ holds
$$[S_j(\pi), S_j(\pi) + p_j) \subseteq [est_j, lct_j).$$
2. Resource capacity constraint, any $r \in R$ and $t \in [0, H)$ satisfy
$$\sum_{j \in N | t \in [S_j(\pi), S_j(\pi)+p_j)} a_{jr} \leq c_r.$$
3. Precedence constraints. If for a pair of tasks $(i, j) \in N^2$ precedence relation e_{ij} is defined, then the following inequality holds
$$S_i(\pi) + e_{ij} \leq S_j(\pi).$$
4. Disjunctive constraints. If for a pair of tasks $(i, j) \in N^2$ disjunctive relation $g_{ij} = 1$ is defined, then processing intervals of these tasks do not intersect, i.e.
$$[S_i(\pi), S_i(\pi) + p_i) \cap [S_j(\pi), S_j(\pi) + p_j) = \emptyset.$$

This paper is organized as follows. The overview of existing resource-based and precedence relations based propagators is presented in section 2. In sections 3 and 4, we propose two new ideas for domain propagation. Section 5 presents concluding remarks.

2 State of the art

There is a wide range of existing constrained programming algorithms which can be applied to find an optimal/suboptimal solution for RCPSP or to make the problem easier to solve. Constraint programming is widely used to solve scheduling problems, including RCPSP. The most comprehensive surveys can be found [2], [3] and [4]. This research focuses on constraints based on resource usage, precedence and/or disjunctive relations.

The term of *disjunctive graph* was proposed in [5]. Algorithms to solve scheduling problems using disjunctive graphs were presented in [6].

The term *compulsory part* of a task was proposed in [7]. In [8], [9] and [10], compulsory parts were used to calculate *time-tables, resource profiles* and *resource histograms.* In papers [11], [12] and [13] time tabling sweep algorithms were presented to adjust task domains. In [14], the time tabling algorithm was improved by solving the rectangle placement problem. The best theoretical complexity of time table sweep algorithms were presented by Gay et. al. [15].

Resource-based edge-finding and extended-edge-finding domain propagation algorithms were proposed in [16], [17], [18] and [19], respectively.

One of the advantages of constraint programming approach is the possibility to combine different algorithms for a more efficient propagation. In [20], [21] and [22], the synergy of Edge-Finding, Extended Edge Finding and Time Tabling algorithms were used to obtain very good results.

To present new propagation ideas, we need the following notations: $[est_j, lct_j)$ is a processing *domain* for task $j \in N$. In this domain, some time intervals can be forbidden for processing task j.

Note that $ect_j = est_j + p_j$ is the earliest possible completion time and $lst_j = lct_j - p_j$ is the latest possible start time. If $[lst_j, ect_j) \neq \emptyset$, then interval $[lst_j, ect_j)$ is a *compulsory part* of task j (firstly defined in [7]).

3 Forbidden start intervals

Suppose that we found a *resource profile* using compulsory parts with the algorithm suggested by Fox [9] or alternatively we calculated $c_r^n(t)$ – the highest possible amount of resource $r \in R$ which can be used by non-compulsory parts of tasks using the algorithm presented in [23]. Then if for any t in domain of j, such that $t \notin [lst_j, ect_j)$ there is not enough resource $r \in R$ to process task j, i.e. $t \in [est_j, lct_j) \setminus [lst_j, ect_j)$ and $c_r^n(t) < a_{jr}$, then it isnot possible to start task j in interval $(t - p_j, t]$ (Fig. 1). The set of all such intervals is denoted by P_j, and $P = \cup_{j \in N} P_j$.

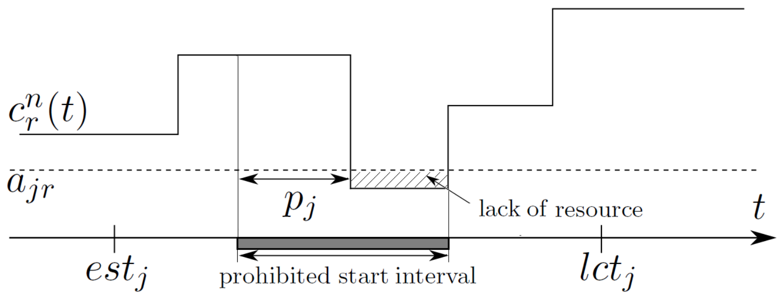

Fig. 1. The lack of resource r indicated a start interval for task j.

3.1 Detection of forbidden start intervals using resource capacity

Forbidden start intervals can be found in the following way. For each resource $r \in R$, we iterate breakpoints t of function $c_r^n(t)$. For all tasks $j \in N$, we check if $t \in [est_j, lct_j) \setminus [lst_j, ect_j)$ and $c_r^n(t) < a_{jr}$, and if it is the case, we add interval $[t - p_j, t' - p_j)$ into the set of forbidden start intervals for task j. We iterate m breakpoints for m resources and for each breakpoint we check $O(1)$ inequalities for n tasks. Therefore, the total complexity of the algorithm is $O(rmn)$.

Note that forbidden start intervals are more informative than task domains. In figure 2, there is an example for task j with a non-empty forbidden start interval while the domain of task is $[0, H)$. It is not possible to start task j in interval $[t_0 - p_j, t'_0)$ but it can be processed at any moment of time $t \in [0, T)$.

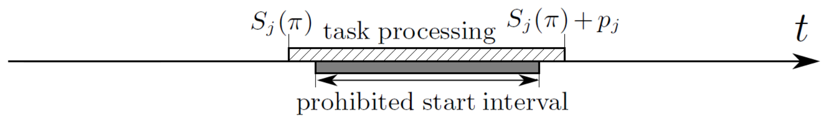

Fig. 2. There is an interval of forbidden start, but task can be processed at any moment of time $t \in [0, T)$.

3.2 Detection of forbidden start intervals using precedence relations

Suppose that there is a pair of tasks $(i, j) \in N^2$ with precedence relations e_{ij} and e_{ji}. Then, the set of forbidden start intervals for task j can be enriched using the set of forbidden start intervals for task i. Precedence relations with time lags imply $S_i + e_{ij} \leq S_j$, and $S_j + e_{ji} \leq S_i$, which leads to

$$S_i + e_{ij} \leq S_j \leq S_i - e_{ji}.$$

Then if there is an interval $[t_0, t'_0)$ in which task i cannot be started, then task j cannot be started in interval $(t_0 - e_{ji}, t'_0 + e_{ij})$ (Fig. 3).

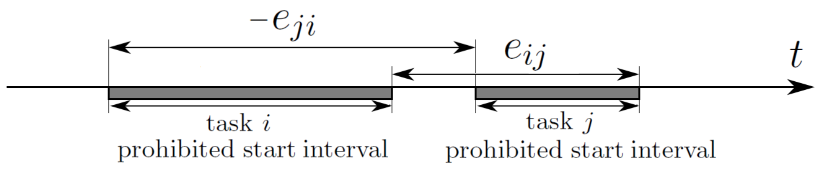

Fig. 3. Construction of forbidden start intervals using precedences with time lags.

4 Disjunctive function

In paper [5], the disjunctive graph was defined. We use a disjunctive function $g_{ij}(t)$ defined on interval $[0, T)$. If tasks i and j can be processed simultaneously at time t, $g_{ij}(t) = 0$, otherwise $g_{ij}(t) = 1$. This formulation generalizes the concept of the disjunctive graph where $g_{ij}(t) = 1$ for each edge $[i, j]$.

The following techniques are used for calculating disjunctive functions.

1. **Intervals of forbidden processing.** If at time t at least one task i or j cannot be processed, then $g_{ij}(t) = 1$.
2. **Resource capacity.** If for any time t and resource $r \in R$ holds
$$a_{ir} + a_{jr} > c_r,$$
then $g_{ij}(t) = 1$.
3. **Resource capacity with non-compulsory usage function.** If for any $t \notin [lst_i, ect_i) \cup [lst_j, ect_j)$ and resource $r \in R$ the following inequality holds
$$a_{jr} + a_{ir} > c_r^n(t),$$
then $g_{ij}(t) = 1$.
4. **Input.** New disjunctive functions can be also defined by input.

Now we will show how disjunctive functions can be used for constraint propagation.

4.1 Criterion for problem unsolvability showed by disjunctive functions

If for any $(i, j) \in N^2$ such that $g_{ij}(t) = 1$ and $t \in [lst_i, ect_i) \cap [lst_j, ect_j)$, then there is no feasible schedule for this input. Proof: each task has to be processed in its compulsory part, therefore both tasks i, j have to be processed at time t, which contradicts $g_{ij}(t) = 1$.

4.2 Generation of new precedence relations using disjunctive functions

Now let us show how disjunctive function can be used for generating new precedence relations. If for two tasks $i, j \in N$ and moment of time t holds $g_{ij}(t) = 1$, and $est_i < t < est_i + p_i$, $lct_j - p_j < t < lct_j$. Then any feasible schedule satisfies the following statements

1. If i starts before t, then the completion time of j is less than t.
2. If completion time of j is greater than t, then i starts after t.

This means that for tasks i and j, new precedence relation with time lag can be defined (Fig. 4)
$$e_{ji} = \min\{r_i - t - p_j, t - d_j + p_j\}.$$

5 Conclusion

Constraint programming approach is a very powerful tool for solving scheduling problems. One of the strengths of this method is the possibility of combining different propagators to tighten task domains and to increase the search speed. In this work, we present new propagators based on calculation of forbidden start intervals and disjunctive functions that generalize the concept of disjunctive graphs. Further efforts on the integration of the proposed approaches with existing constraint propagators into the preprocessing and search algorithms are required in order to achieve their full performance.

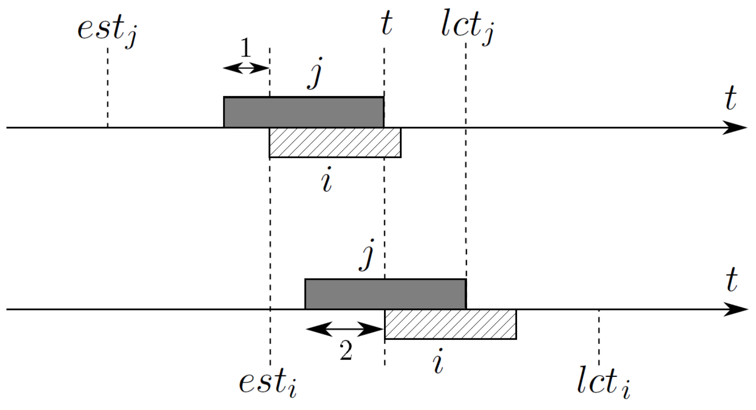

Fig. 4. Generation of new precedence relations using disjunctive functions. e_{ij} is not less than the minimum of values calculated for cases 1 and 2.

References

1. Garey, M., Johnson, D.: Complexity results for multiprocessor scheduling under resource constraints. SIAM Journal on Computing 4 (4), 397–411 (1975).
2. Baptiste, P., Le Pape, C., Nuijten W.: Constraint-Based Scheduling. Kluwer Academic Publishers (2001).
3. Laborie, P.: Algorithms for propagation of resource constraints in AI planning and scheduling: Existing approaches and new results. Artificial Intelligence 143, 151–188 (2003).
4. Vilim, P. Global Constraints in Scheduling. PhD thesis. Charles University in Prague, Faculty of Mathematics, Physics, Department of Theoretical Computer Science, and Mathematical Logic. url: http://vilim.eu/petr/disertace.pdf, (2007).
5. Roy, B., Sussman, M. A.: Les problemes d'ordonnancement avec contraintes disjonctive. Tech. rep. Note DS No.9 bis. SEMA, Paris (1964).
6. Carlier, J., Pinson, E.: An Algorithm for Solving the Job-Shop Problem. In: Management Science 35.2, 164–176 (1989).
7. Lahrichi, A.: Scheduling: the Notions of Hump, Compulsory Parts and their Use in Cumulative Problems. In: C.R. Acad. Sc. Pairs 294, 209–211, (1982).
8. Le Pape, C.: Des syst mes d'ordonnancement exibles et opportunistes. PhD thesis. Universite Paris XI (1988).
9. Fox, B. R.: Non-chronological scheduling. In: Proceedings of AI, Simulation and Planning in High Autonomy Systems, pp. 72–77 (1990).
10. Caseau, Y., Laburthe F.: Cumulative Scheduling with Task Intervals. In: Proceedings of the Joint International Conference and Symposium on Logic Programming, 363–377 (1996).
11. Beldiceanu, N., Carlsson, M. . Sweep as a Generic Pruning Technique Applied to the Non-overlapping Rectangles Constraint. In: Proceedings of the seventh International Conference on Principles and Practice of Constraint Programming, 377–391 (2001).

12. Beldiceanu, N., Carlsson, M.: A new multi-resource cumulatives constraint with negative heights. In: Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming, 63–79 (2002).
13. Letort, A., Beldiceanu, N., Carlsson, M.: A scalable sweep algorithm for the cumulative constraint. In: Proceedings of the 18th International Conference on Principles and Practice of Constraint Programming, 439–454 (2012).
14. Beldiceanu, N., Carlsson, M., Poder, E.: New filtering for the cumulative constraint in the context of non-overlapping rectangles. In: Proceedings of the 5th International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 21–35 (2008).
15. Gay, S., Hartert, R., Schaus, P.: Simple and Scalable Time-Table Filtering for the Cumulative Constraint. In: Proceedings of the 21st International Principles and Practice of Constraint Programming CP, 149–157 (2015).
16. Nuijten, W.: Time and Resource Constrained Scheduling. PhD thesis, Eindhoven University of Technology (1994).
17. Vilim, P.: Edge finding filtering algorithm for discrete cumulative resources in O(kn log n). In: Proceedings of the 15th International Conference on Principles and Practice of Constraint Programming, 802–816 (2009).
18. Kameugne, R., Fotso, L., Scott, J.: A Quadratic Edge-Finding Filtering Algorithm for Cumulative Resource Constraints. In: Proceedings of the Seventeenth International Conference on Principles and Practice of Constraint Programming CP, 478–492 (2011).
19. Mercier, L., Van Hentenryck, P.: Edge finding for cumulative scheduling. INFORMS Journal on Computing 20(1), 143–153 (2008).
20. Vilim, P.: Timetable edge finding filtering algorithm for discrete cumulative resources. In: Proceedings of the 8th International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 230–245 (2011).
21. Ouellet, P., Quimper, C.: Time-Table Extended-Edge-Finding for the Cumulative Constraint. In:Proceedings of the Nineteenth International Conference on Principles and Practice of Constraint Programming CP, 562–577 (2013).
22. Schutt, A., Feydy, T., Stuckey, P.J.: Explaining time-table-edge-finding propagation for the cumulative resource constraint. In: Proceedings of the 10th International Conference on Integration of AI and OR Techniques and Constraint Programming for Combinatorial Optimization Problems (2013).
23. Arkhipov D., Battaïa O., Lazarev A., Tarasov G., Tarasov, I. : New task domain propagators with polynomial complexity for Resource-Constrained Project Scheduling Problem. In: Proceedings of IX International Conference on Optimization and Applications, DEStech Proceedings (2018), in print.