# SAT-based optimisation for the RCPSP/t

Jelle Pleunes (J.G.Pleunes@student.tudelft.nl), supervisor: Emir Demirović

## 1 What is the RCPSP/t?

**R**esource-**C**onstrained
**P**roject **S**cheduling **P**roblem
with **T**ime-dependent
resource capacities and requests

For example, you are given:

- A set of activities $A = \{1, 2, 3, 4\}$, where each $i \in A$ has a duration.
- A set of resources $R = \{1, 2\}$.

There are precedence constraints:

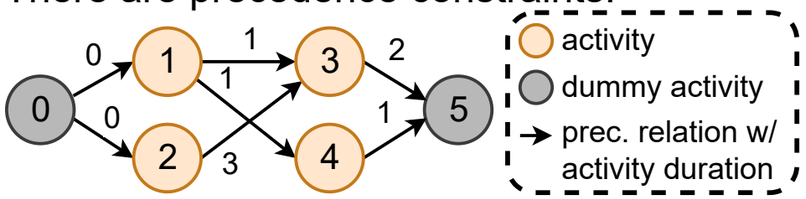


**Figure 1:** Precedence graph: an activity can only start once all predecessors are finished.

And there are resource constraints:

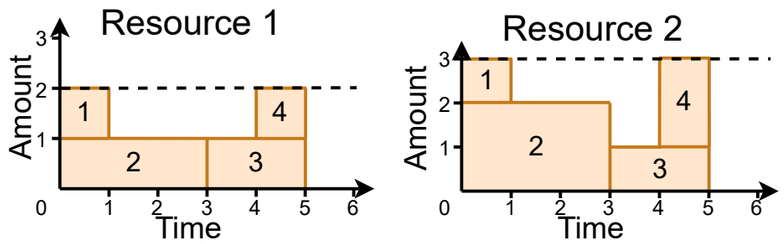


**Figure 2:** RCPSP resource constraints. Capacity (dotted line) may not be exceeded by requests of acitivties (boxes).

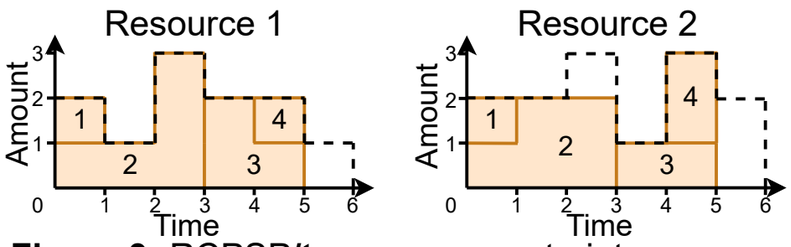


**Figure 3:** RCPSP**/t** resource constraints. Capacities and requests are now time-dependent.

**Goal: assign start times, minimising total project duration. (NP-hard!)**

## 2 Practical applications?

Model & schedule industrial processes [1]:

Production scheduling | Medical research projects | More...

## 3 How to find optimal solutions?

1. Encode (reduce) RCPSP/t into Boolean satisfiability problem (SAT), or one of its exensions such as MaxSAT[1] or SMT[2].
2. Exact solver solves encoded problem.

Existing research [2] used SMT encoding.
+ Allows you to simply write linear inequalties for precedence constraints.
- SMT solvers are complex.

This work: **new** SAT (& MaxSAT) encoding.
- Slightly more variables/clauses needed for encoding (< 2% increase).
+ SAT solvers are less complex.

**Research question:**
**Is a SAT encoding efficient for solving?**

[1]Maximum satisfiability, [2]satisfiability modulo theories

## 4 Results

| $n = 30$ | $t_{total}$ | $\#c$ | $\Delta_{LB}$ |
|---|---|---|---|
| SMT | 0.93 | <u>2875</u> | <u>0.00%</u> |
| SAT | 1.44 | 2845 | 0.02% |
| MaxSAT | 1.64 | 2843 | 0.02% |

| $n = 120$ | $t_{total}$ | $\#c$ | $\Delta_{LB}$ |
|---|---|---|---|
| SMT | 35.13 | 1758 | 6.71% |
| SAT | 32.28 | <u>1854</u> | 4.25% |
| MaxSAT | 42.60 | 1822 | <u>3.18%</u> |

**Figure 4:** Average performance on test instances. $n$ activities per instance, 2880 instances (L) and 3600 instances (R). $t_{total}$: execution time (s), $\#c$: num. proven optimal/infeasible, $\Delta_{LB}$: distance from good known solution. Best values <u>underlined</u>.

## 5 Main conclusion

SAT and MaxSAT approaches are efficient; performance of both scales better than the SMT approach for larger problem instances.

## 6 Limitations

All studied solving approaches used pseudo-Boolean encoding for resource constraints, instead of state-of-the-art pseudo-Boolean at-most-one encoding [2].
$\rightarrow$ Performance could be improved.

Only one solver measured per encoding.
$\rightarrow$ Different solvers may result in different performance; this work makes it possible to use many different solvers.

## 7 Future work

New SAT encoding can be used.
SAT solvers are less complex, making them more suitable for implementing heuristic augmentations, which could further improve performance.

## References

[1] S. Hartmann. “Time-varying resource requirements and capacities”. In: *Handbook on Project Management and Scheduling*. Vol. 1. Springer, 2015, pp. 163–176.

[2] M. Bofill et al. “SMT encodings for resource-constrained project scheduling problems”. In: *Computers & Industrial Engineering* 149 (2020).