Resource-constraint project scheduling with task preemption and setup times by Boolean satisfiability encoding and satisfiability (SAT) solver

1

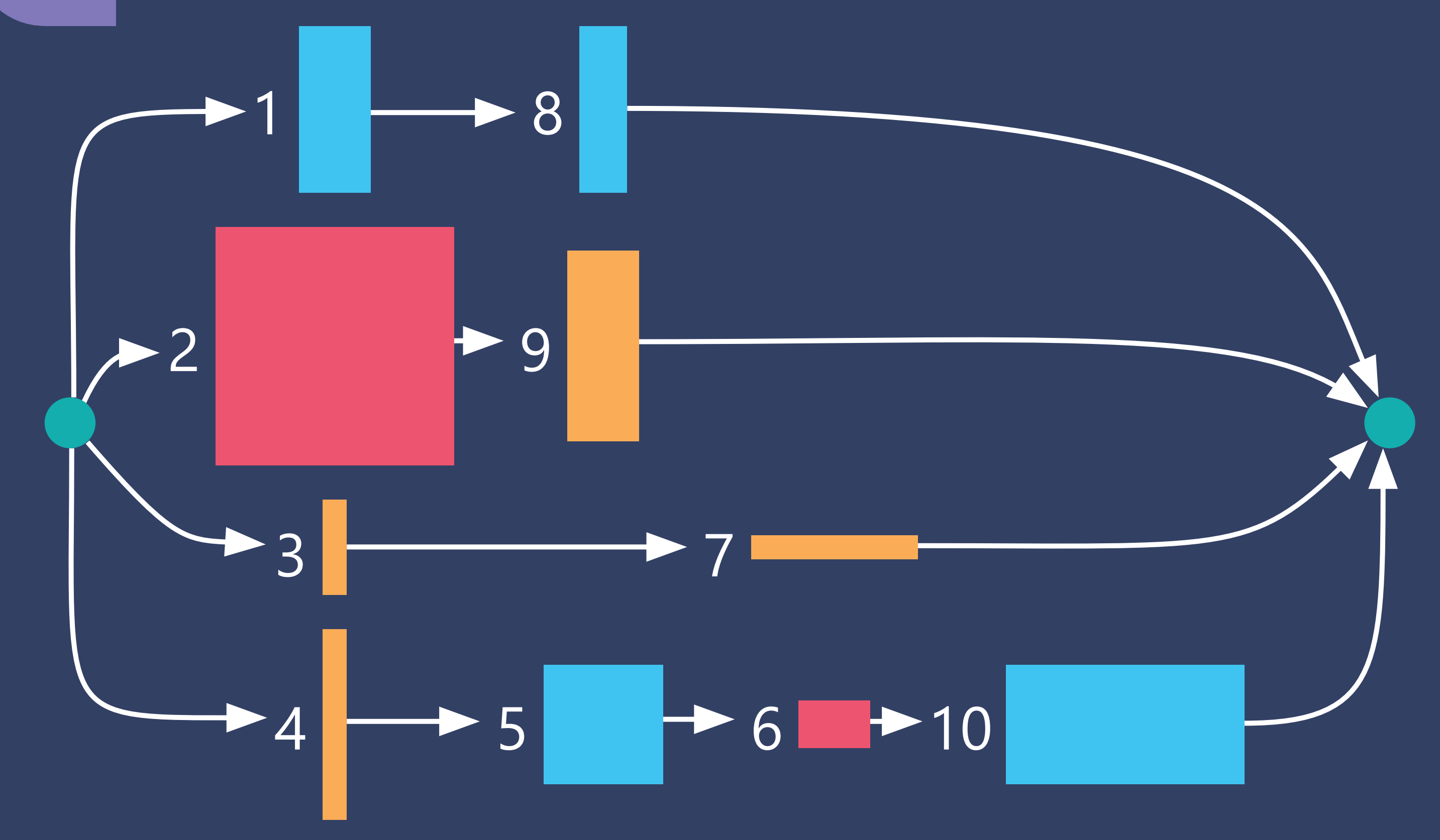
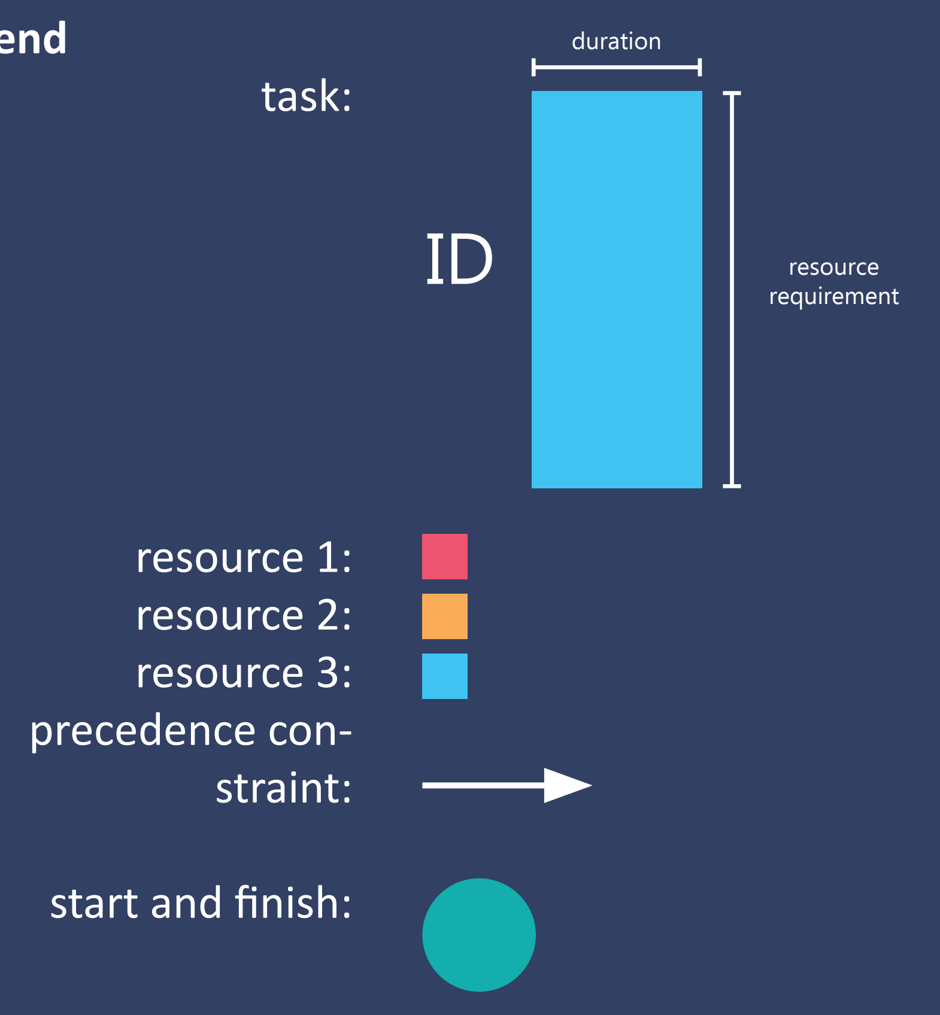


Figure 1: Example project task network
Source: Adapted from [1]

Legend



2

Background

Resource constraint project scheduling problem (RCPSP)
- a project has multiple tasks
- each task needs an amount of specific resource
- each task has a duration
- tasks have precedence constraints
- all tasks must be scheduled
- the objective is to try and find the shortest makespan (makespan = total schedule time)

Preemption and setup times (PRCPSP-ST)
- a task can be put on hold before it finishes
- each preemption is penalized (setup time)
- preemption must result in a shorter schedule

Figures 2 and 3 show a reduction of the schedule duration by allowing preemption

By:
Jasper Vermeulen, j.vermeulen-1@student.tudelft.nl

Supervisor:
Emir Demirović

At:
23/06/2022, Mekelweg 5, 2628 CD Delft

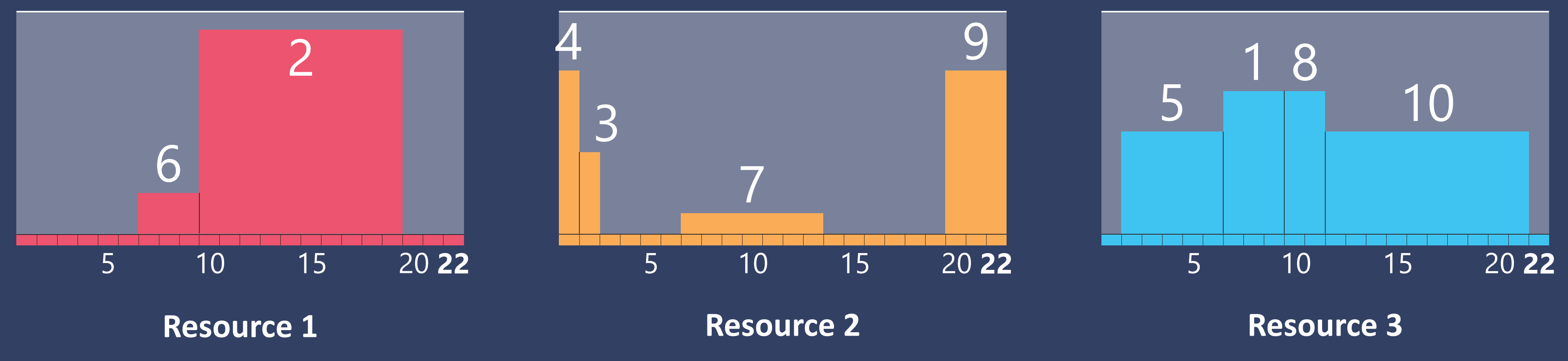


Figure 2: Shortest possible schedule of duration 22 without preemption
Source: Adapted from [1]

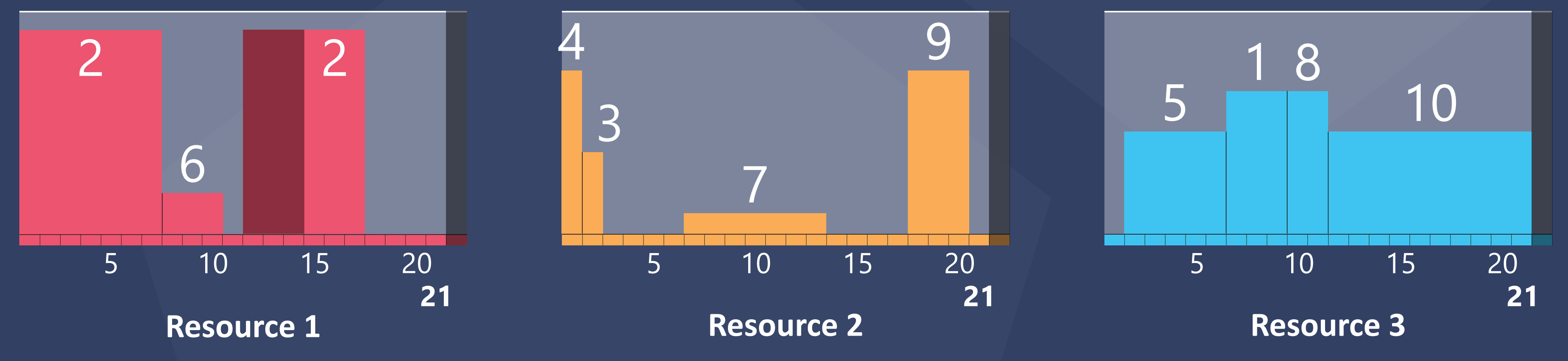


Figure 3: A shorter possible schedule of duration 21 with preemption of task 2
Source: Adapted from [1]

3

Motivation, related work and research question

Scheduling provides a hard problem to solve algorithmically and with applications in most production and logistic industries new solutions can provide much profit.

The resource-constrained project scheduling problem (RCPSP) is no exception.
Multiple different variations are possible and preemption with setup times is one of them.

To start out preemption did not seem to provide makespan reductions when an exact branch-and-bound procedure was applied [2].
Still, research continued and the meta-heuristic linear-programming algorithms were adapted to succesfully deal with preemption [3, 4].
When branch-and-bound was revisited with the addition of fast-tracking (allowing task segments to be processed parallel) a more significant makespan reduction was found to be possible. [5]
With a genetic algorithm applied on a combined model of preemption and setup times, it was shown that makespan reduction was indeed possible contrary to previous research [1].

A satisfiability (SAT) solver has been used as a part of the genetic algorithm.
It could be used as a complete solution when RCPSP with preemption and setup times if it can be encoded as Boolean satisfiability logic.

This paper proposes a SAT encoding for the RCPSP while also allowing tasks to be split.
A SAT solver is used to solve the encoded problem instances and compared with the results of a heuristic algorithm.
The intention is to show better results in makespans with the SAT solver and provide proof of optimality with the main research question:

"Can a satisfiability encoding of PRCPSP-ST models solved on a SAT solver be used to reduce the average makespan of the resulting schedule compared to a heuristic algorithm when run for an equal amount of time?

4

Experimental setup and results

To answer the research question a tweaked version of the iterated greedy algorithm is compared to a SAT encoding and solver.
Experiments were run on the high-performance computing cluster at the Delft University of Technology with three datasets: J30, DC1 and RG30.
The datasets each had 480 instances with projects ranging from 10 - 30 tasks.
For each instance in the dataset, the heuristic and SAT approaches were run for 60s CPU time.

The first results table shows:
- average percentage deviation (%Dev) from best-known solutions for heuristic and SAT
- percentage of instances with reduced makespan compared to known optimal makespans
- average percantage deviation (%Dev) from best-known solutions for improved instances

Dataset	%Dev heur.	%Dev SAT	%Imp heur.	%Imp SAT	%Dev heur.	%Dev SAT
DC1	0.13 %	0.1 %	5.9 %	12 %	-3.8 %	-4.4 %
J30	2.4 %	1.2 %	1.6 %	5.6 %	-3.2 %	-3.7 %
RG30	5.6 %	10.5 %	0.49 %	1.6 %	-0.8 %	-3.0 %

The second results table shows:
- percentage of instances solved by SAT solver
- percentage of solved instances that were solved with an optimal schedule
- percentage of instances that were solved by SAT that improved the heuristic result
- percentage of instances that were solved by SAT that matched the heuristic result
- percentage of instances that were solved by SAT where the heuristic found a better result

Dataset	%Satisfied	%Optimal	%SAT outperformed	%Equal	%Heuristic outperformed
DC1	75.2 %	67.5 %	17 %	73 %	10 %
J30	28.8 %	81.9 %	36 %	56 %	8 %
RG30	24.0 %	9.0 %	23 %	7 %	70 %

5

Discussion & Conclusion

- The algorithms produced results close to the best-known solutions for the DC1 and J30 datasets
- For the RG30 dataset, the results still deviate 5% from the best solutions and are considered inadequate for further comparison
- The number of improved instances and a deviation of around -4% show a good motivation for allowing preemption when an absolute minimum makespan is required even when tasks will be split
- Depending on the dataset the proposed SAT encoding can find solutions in between 29% and 75% of the instances. This number might already be good enough for applications but improvements can be made
- When the SAT solver does find a solution the results match or outperform the heuristic approach in most cases (not considering the RG30 dataset which wasn't solved adequately)
- The setup times had little impact on the reduction of makespans when tasks would be preempted

Comparing the approaches shows the benefit of going for a SAT solver method if makespan reduction is necessary enough to sacrifice intermediate results in case of a timeout. This claim is made only stronger by the proofs of optimality that only an exact method like SAT can provide.

[1] M. Vanhoucke and J. Coelho, "Resource-constrained project scheduling with activity splitting and setup times," *Computers & Operations Research*, vol. 09, pp. 230–249, 2019. [Online]. Available: https://doi.org/10.1016/j.cor.2019.05.004
[2] E. L. Demeulemeester and W. S. Herroelen, "An efficient optimal solution procedure for the preemptive resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 90, no. 2, pp. 334–348, 1996. [Online]. Available: https://doi.org/10.1016/0377-2217(95)00358-4
[3] J. Damay, A. Quilliot, and E. Sanlaville, "Linear programming based algorithms for preemptive and nonpreemptive rcpsp," *European Journal of Operational Research*, vol. 182, no. 3, pp. 1012–1022, 2007. [Online]. Available: https://doi.org/10.1016/j.ejor.2006.09.052
[4] F. Ballest´ın, V. Valls, and S. Quintanilla, "Preemption in resource-constrained project scheduling," *European Journal of Operational Research*, vol. 189, no. 3, pp. 1136–1152, 2008. [Online]. Available: https://doi.org/10.1016/j.ejor.2006.07.052
[5] M. Vanhoucke, "Setup times and fast tracking in resource-constrained project scheduling," *Computers & Industrial Engineering*, vol. 54, no. 4, pp. 1062–1070, 2008. [Online]. Available: https://doi.org/10.1016/j.cie.2007.11.008