Exploring the Effect of Explanations for Energetic Reasoning

Konstantin Kamenov
k.kamenov@student.tudelft.nl

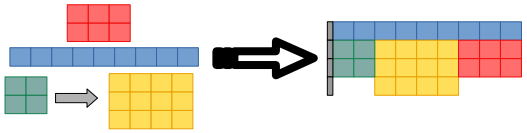
Supervisors: Imko Marijnissen
Responsible professor: dr. Emir Demirović

TU Delft

1. Problem Definition (RCPSP)

Find the shortest viable schedule, given:

- Several limited resources
- A set of tasks (Start times, Duration, Resource consumption)
- Precedence relations



2. Background

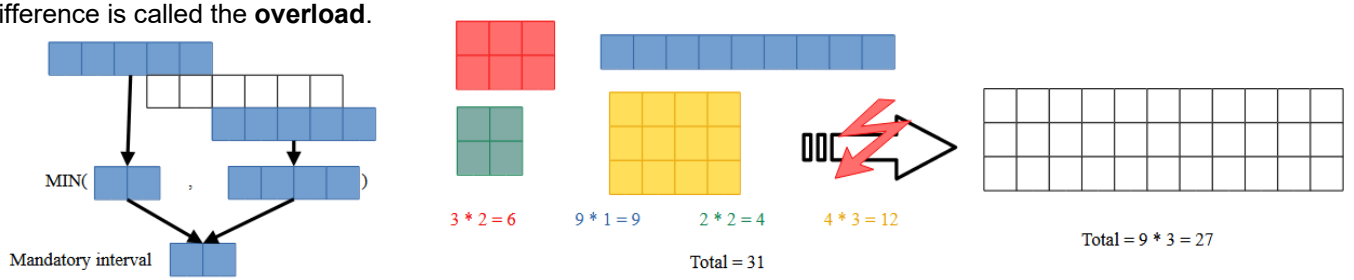
Constraint programming - technique for solving NP-hard optimization by utilizing backtracking and propagators.

Conflict analysis[1, 2] - technique which allows us to find the actual source of a conflict and backtrack more efficiently.

Explanations - clauses which explain why conflict and propagations happen. The more general they are, the more effective conflict analysis will be.

Cumulative constraint - for a single resource, at no time we end up using more resource than available.

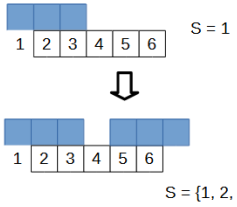
Energetic reasoning[3] - a method for propagating this constraint by reasoning about the available energy in an interval and the mandatory energy of tasks in the interval. If the mandatory energy is more, we have a conflict and the difference is called the **overload**.



3. Main contribution

Not much exploration has been done into creating more general explanations for the energetic reasoning propagator.

Initial bounds relaxations: Find new bounds for the task which are as wide as possible without affecting its mandatory interval within an interval.



Reducing the overload: move/remove tasks, such that the new placement has lower overload, but still causes a conflict.

Proposed strategies are:

1. Shift – Try to "move out" the task with lowest resource consumption one time unit at a time
2. Greedy – Remove the task with lowest energy from the conflict
3. Knapsack – Use knapsack in order to remove tasks, obtaining the conflict, which is most likely to happen

4. Results

Setup - 65 J120 instances, 55 pack instances, 10 minutes timeout

Primary metric – number of conflicts (conflicts gain) with lower values being better

Initial bounds relaxation – more than 25% reduction

Overload reduction – between 1% and 5% reduction

	Conflicts gain		Runtime gain		Backtrack gain		LBD gain	
Solution	J120	pack	J120	pack	J120	pack	J120	pack
naive	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
relaxations	0.688	0.745	0.805	0.773	1.254	1.036	1.295	1.109

	Conflicts gain		Runtime gain		Backtrack gain		LBD gain	
Solution	J120	pack	J120	pack	J120	pack	J120	pack
relaxations	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
shift	0.965	0.988	0.988	0.999	1.026	1.001	0.981	0.993
greedy	0.964	0.984	0.989	0.988	1.029	1.001	0.975	0.992
knapsack	0.955	0.984	0.974	0.985	1.026	1.001	0.976	0.991

5. Conclusion

Bounds relaxations provide a reduction in the number of conflicts of at least 25%.

Overload reduction provides a more limited decrease of less than 5%.

Future work:

- Explore new overload strategies.
- Check the effect of these explanations on faster ER algorithms.
- Look for faster implementations of the knapsack approach

References:

1. Thibaut Feydy and Peter J. Stuckey. Lazy clause generation reengineered. In Ian P. Gent, editor, Principles and Practice of Constraint Programming- CP 2009, pages 352–366, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.
2. Olga Ohrimenko, Peter J. Stuckey, and Michael Codish. Propagation via lazy clause generation. Constraints, 14(3):357–391, Sep 2009. doi:10.1007/s10601-008-9064-x.
3. Ph Baptiste, Claude Le Pape, and Wim Nuijten. Satisfiability tests and time-bound adjustments for cumulative scheduling problems. Annals of Operations research, 92(0):305–333, 1999.