Evaluating the impact of explanations on the performance of an Edge-Finding propagator for the Disjunctive Constraint

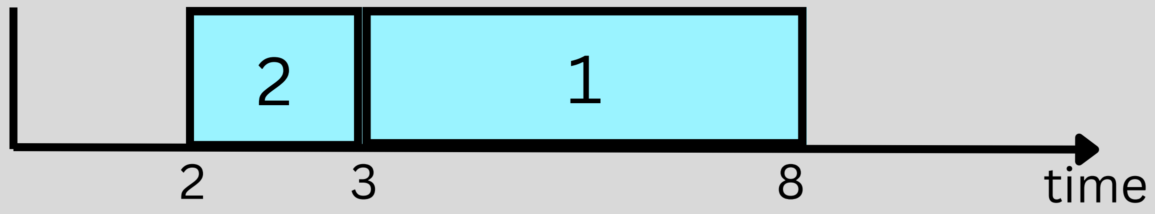
Author: Radu Andrei Vasile
Email: r.a.vasile-1@student.tudelft.nl
Supervisor: Imko Marijnissen
Responsible professor: Dr. Emir Demirović

1. Disjunctive

Schedule n tasks on one machine such that no tasks overlap

- Task 1: start_times = [0,3], duration = 5
- Task 2: start_times = [2,4], duration = 1

Example of an earliest schedule with disjunctive constraint:



Edge-finding rule: est_(Ω+i) + p_(Ω+i) > lct_Ω → Ω << i

Update rule: Ω << i → est_i ≥ ect_Ω

Goal: identify pairs (Ω, i), update bounds of tasks

2. Explanations

Reason for which propagator made a decision. Example:

- **Variables**: x ∈ [0,10], y ∈ [0,10]
- **Constraint**: x + y >= 5
- **Explanation:** [x == 1] → [y >= 4]

Used to learn nogoods - Ensure the same conflict never happens again in the future. → **Lazy Clause Generation**

Conflict windows: Explanations for scheduling problems

Idea: extend variable domains as much as possible, with the condition that the solution remains infeasible.

Why? → If the interval is larger, it helps us by learning more general clauses, which can stick around for longer.

3. Research Question

What is the impact on performance of using explanations on an edge-finding propagator for the disjunctive constraint, implemented in a LCG solver (Pumpkin)?

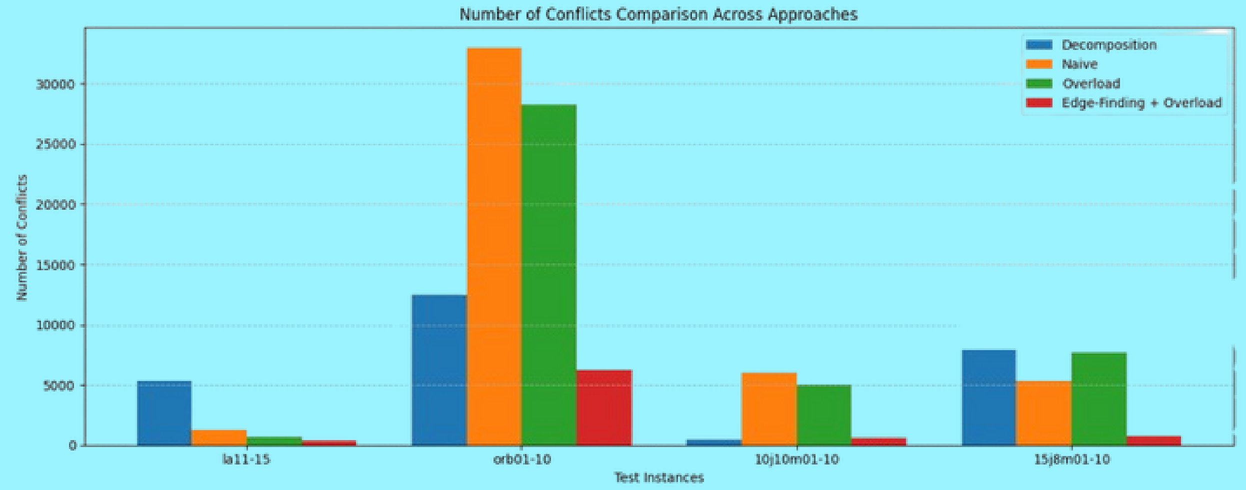
Subquestions:

- What strategies can be used to generate explanations?
- How can edge-finding be adapted to record explanations?

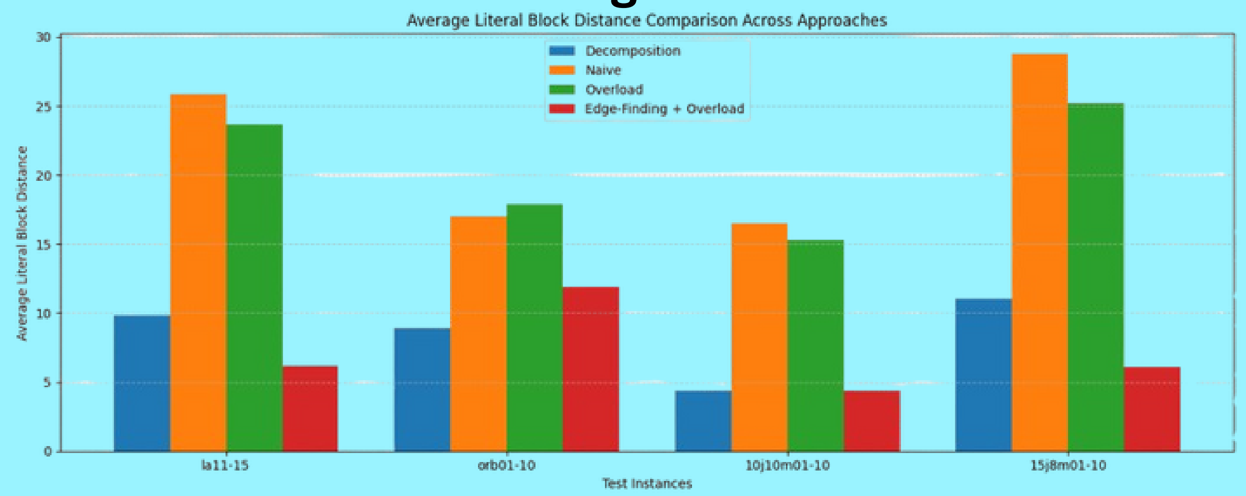
Goal: Explore multiple explanation strategies, benchmark their performance

5. Results

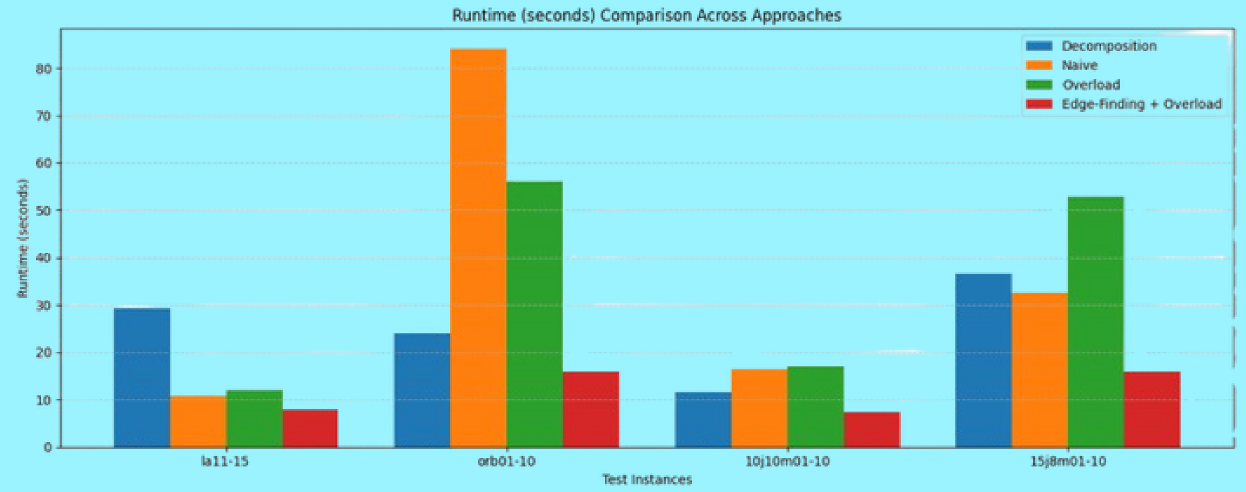
Number of conflicts



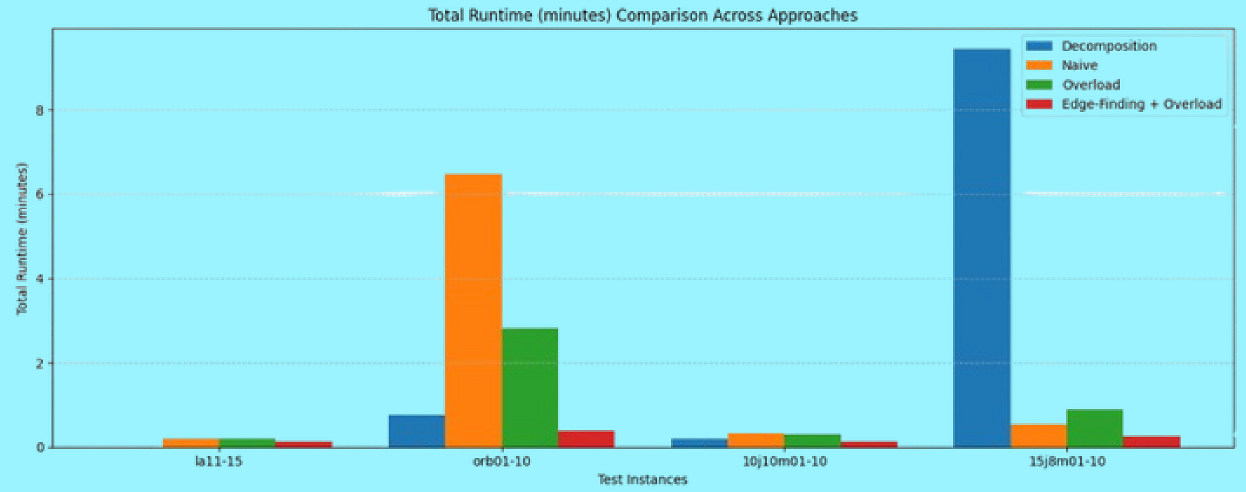
Average LBD



Runtime*



Total Runtime**



*Runtime (in seconds) until optimal solution found, not until finished execution

**Runtime (in minutes) until optimality is proven. On first instance, execution times out for decomposition (>20 minutes) and was removed for scale.

4. Explored variants

- **Decomposition:** Baseline.
- **Naive:** Conjunction of bounds of all variables.
- **Overload:** Lift bounds for variables that cause conflict.
- **Overload + Edge-Finding:** Only consider variables directly responsible for propagation (Ω).

Explanation formulas adapted from previous work [1]

NEW: Extension to Edge-Finding [2] - Modified algorithm to allow generation of explanations in O(n).

6. Conclusions

- Edge-Finding explanations (red) do provide a significant improvement in performance.
- Despite additional complexity, final variant (red) performs, on average, 7 times better than Naive (orange).
- Overload (green) slightly outperforms Naive (orange)
- Decomposition (blue) competes with Naive & Overload.
- Decomposition (blue) struggles at proving optimality

7. Future Work

- Explore/Discover other explanation strategies
- Compare results against other Disjunctive propagators.
- Further investigate surprising results achieved by decomposition.

References

[1] Petr Vilím. Computing explanations for the unary resource constraint. In Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR 2005), volume 3524, pages 396 – 409, 2005.

[2] Petr Vilím. Global constraints in scheduling. Ph.d. thesis, Univerzita Karlova, Matematicko-fyzikální fakulta, 2007. https://dspace.cuni.cz/bitstream/handle/20.500.11956/12252/140038772.pdf.