Half-checking Propagators

Mikael Zayenz Lagerkvist

Magnus Rattfeldt

What if Propagators could remove solutions?

- Propagation incorrect
- Solving incomplete
- ...more propagation?

Propagators are awesome!

- Smart propagation is crucial
- Key to CP success
- Complicated
  - Never remove solutions
  - Propagation is coNP-style problem

What is a propagator?

Function from domains to domains.

1. Local
2. Contracting, $p(d) \subseteq d$
3. Checking, $p(a) = a$ iff $a \in c$
4. Weakly monotonic, $a \in d, p(a) \subseteq p(d)$

What is a half-checking propagator?

Function from domains to domains.

1. Local
2. Contracting, $p(d) \subseteq d$
3. Half checking, if $p(a) = a$ then $a \in c$
4. ~~Weakly monotonic~~

Implications

- Not complete solving
- Lifting incompleteness to propagators
- More than domain consistency

Integration into a system

- Recomputation
- Testing is complicated
- Portfolio solving
  - No-goods are not globally valid

Three general ideas applied to `cost-circuit`

How to design half-checking propagators?

- Dominating solutions
  - No crossing lines
- Heuristic bounds
  - Christofides-Serdyukov bound
- Heuristic deductions
  - one-tree based deductions

Other approaches

- All incomplete methods
  - Local search, LNS,
- Streamliner constraints
  - Propagators vs models
- Belief and cost propagation

Key takeaways

- New idea for incomplete solving
  - Half-checking propagator
- Can be combined with other techniques
- No-goods need to be handled carefully
- github.com/zayenz/half-checking-propagators