Incremental Search for Counterexample-Guided Cartesian Abstraction Refinement

Jendrik Seipp, Samuel von Allmen, Malte Helmert
October 28, 2020

University of Basel
In a Nutshell

• optimal classical planning
• A* search + abstraction heuristic
• counterexample-guided Cartesian abstraction refinement
• bottleneck: find shortest path
• incremental search: 1000x speedup
**CEGAR**

compute initial abstraction

until **TERMINATE()**:

- find shortest path in abstraction
  - **if** there is no path:
    - **return** unsolvable
  - find flaw in path
    - **if** there is no flaw:
      - **return** plan
  - refine abstraction for flaw

**return** abstraction
Example Task

- **load-in-A**
- **unload-in-A**
- **drive**
- **unload-in-B**
- **drive**
- **load-in-B**
Abstraction Refinement

drive, (un)load-in-A, (un)load-in-B
Abstraction Refinement

drive, (un)load-in-A

unload-in-B

load-in-B

drive
Bottleneck: Find Shortest Path

• Dijkstra’s algorithm
• A* search
• search times grow
Bottleneck: Find Shortest Path

- Dijkstra’s algorithm
- A* search
- search times grow
  → incremental search
Incremental Search

• dynamic shortest path
• add/remove transitions
• increase/decrease weights
• fixed set of states
Two-Step Refinement for CEGAR

before splitting $v$

copy $v$

prune transitions
INCREMENTE (Frigioni et al., 2000)

- increasing weights, removing transitions
- shortest path tree
- algorithm:
  - reconnect ancestor states, mark rest dirty
  - run Dijkstra on dirty states
Incremental vs. A* failed times.
Solved Tasks Over Time

Time in seconds vs. solved tasks for two algorithms:
- INC
- A*
CEGAR bottleneck: find shortest paths
- cast as dynamic shortest path problem
- drastic speedup