Optimized Computation of the Additive and FF heuristics

Bachelor Thesis

Cyrill Imahorn

Motivation

- Planning Task are important
- Example: Go
- Planning Task use heuristics
- Fast Downward

Planning Task

- Describes a Planning Problem
- Is a 4-tuple $\Pi = \langle V, I, O, \delta \rangle$:
  - Variables V
  - Initial state I
  - Operators O
  - Goal conditions δ

Operators

- Change a partial assignment of V
- Three main characteristics:
  - Preconditions
  - Effect
  - Cost

Heuristic

- Estimates how “good” a state is
- Simplify planning task
- Delete relaxation heuristics (Additive and FF heuristic)
  - Can be solved in polynomial time
  - Can be solved with a directed graph

Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

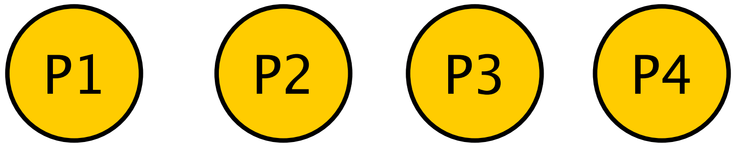
$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

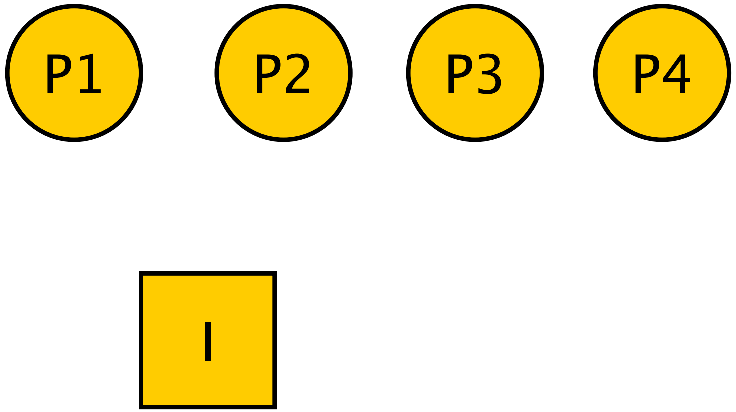


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

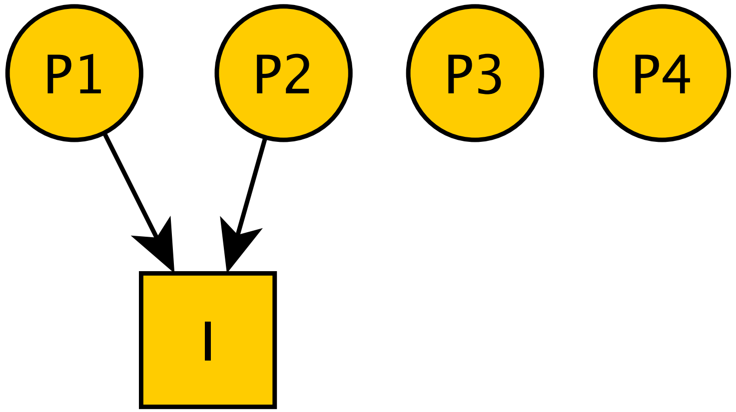


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

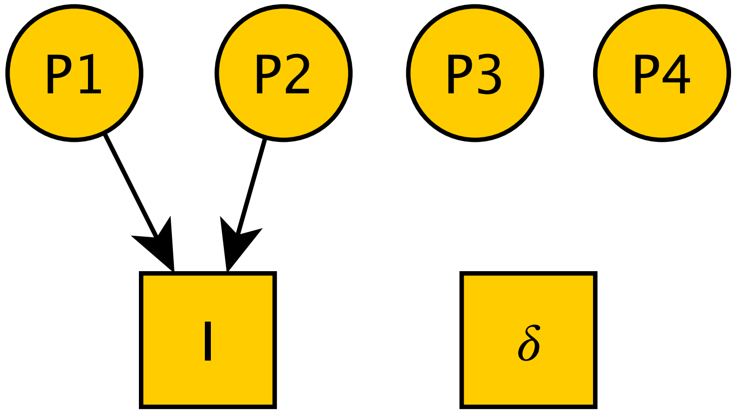


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

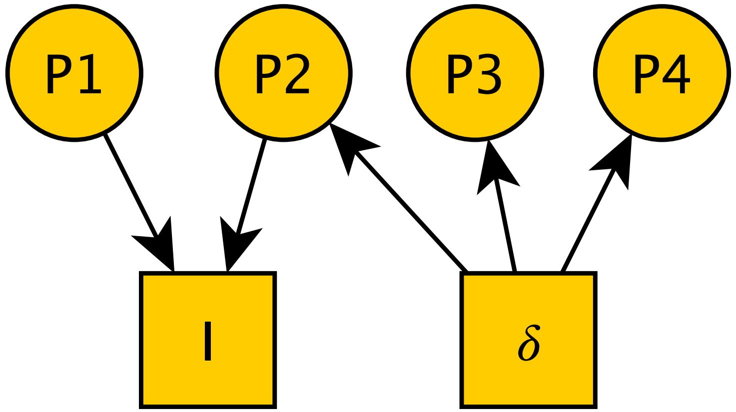


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

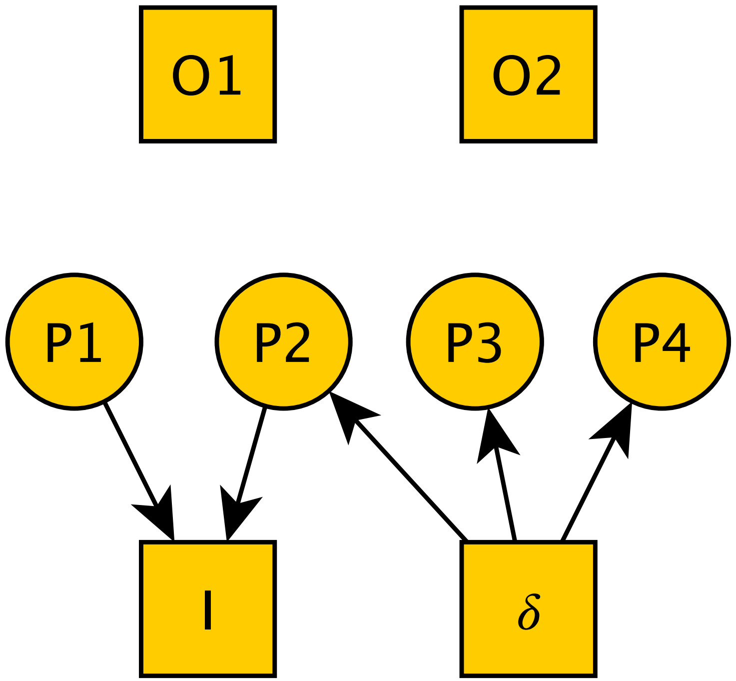


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

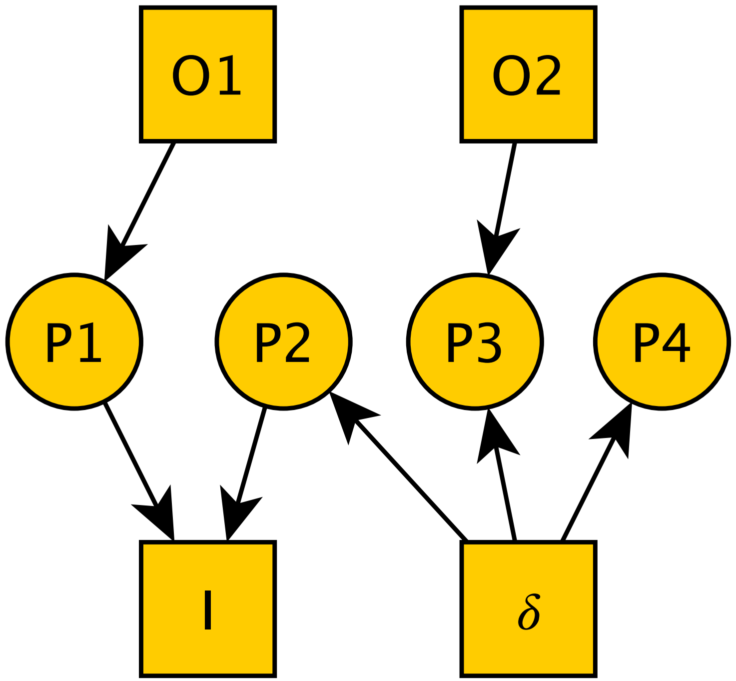


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

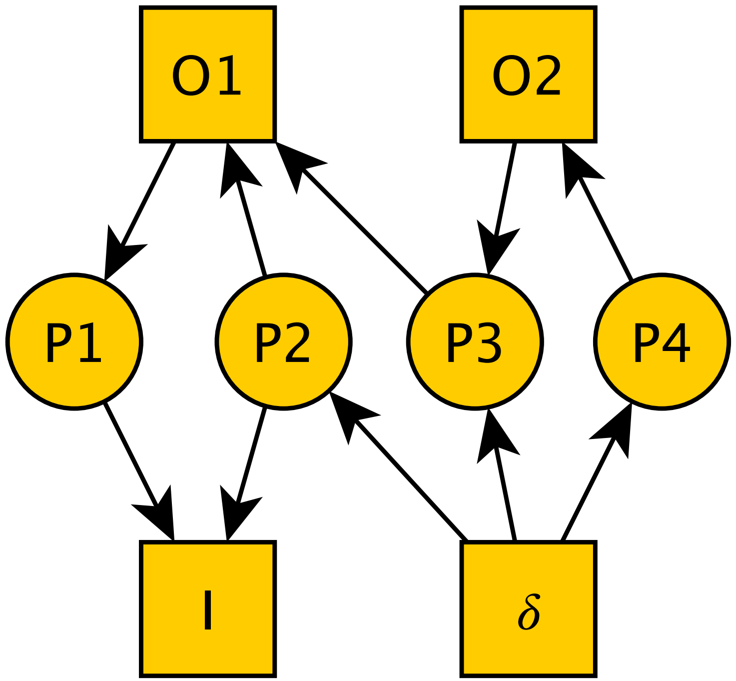


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

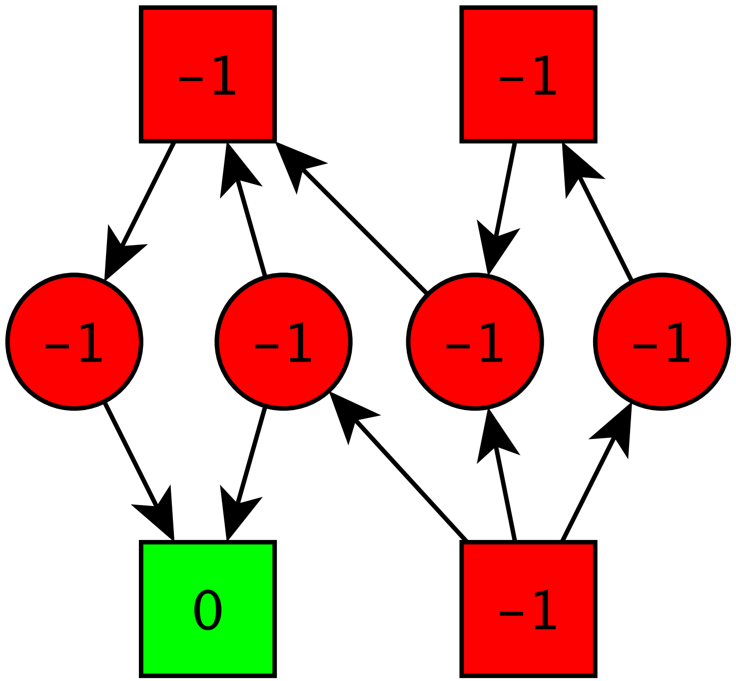


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

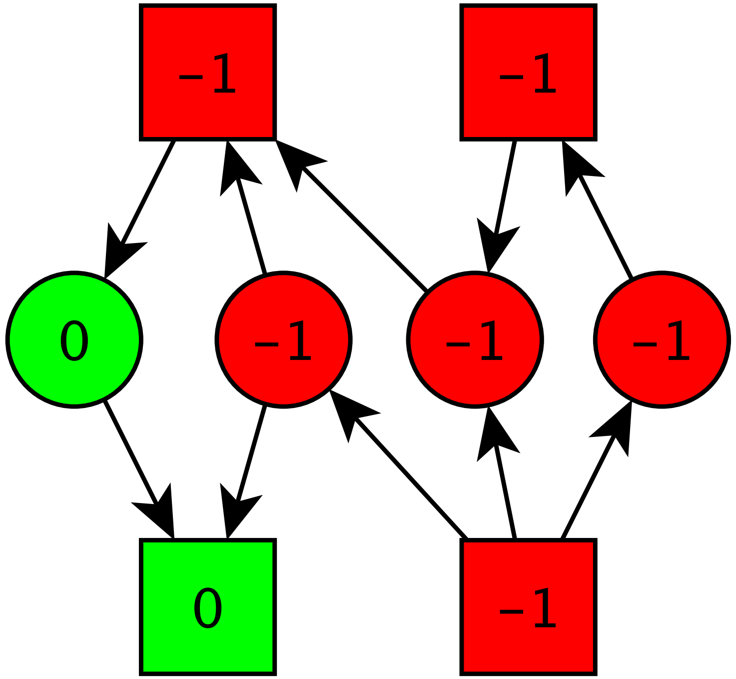


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

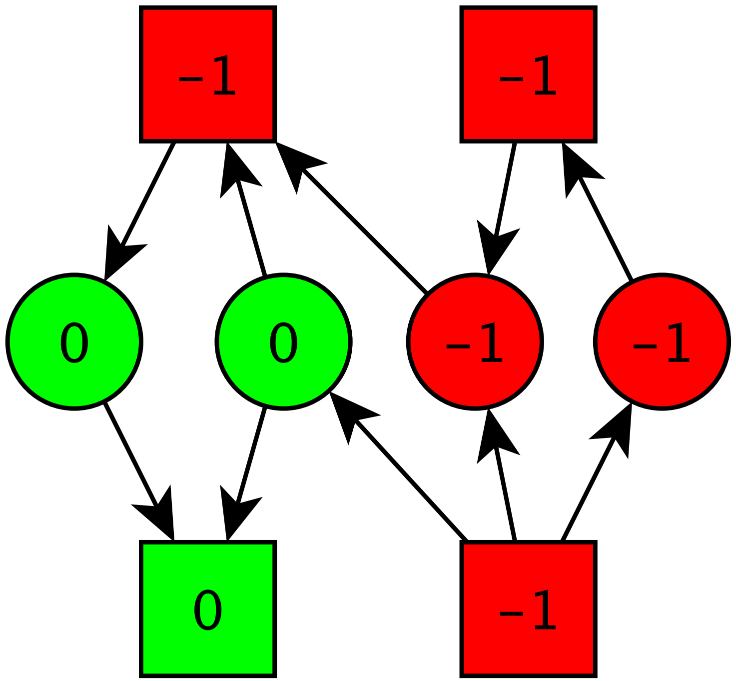


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

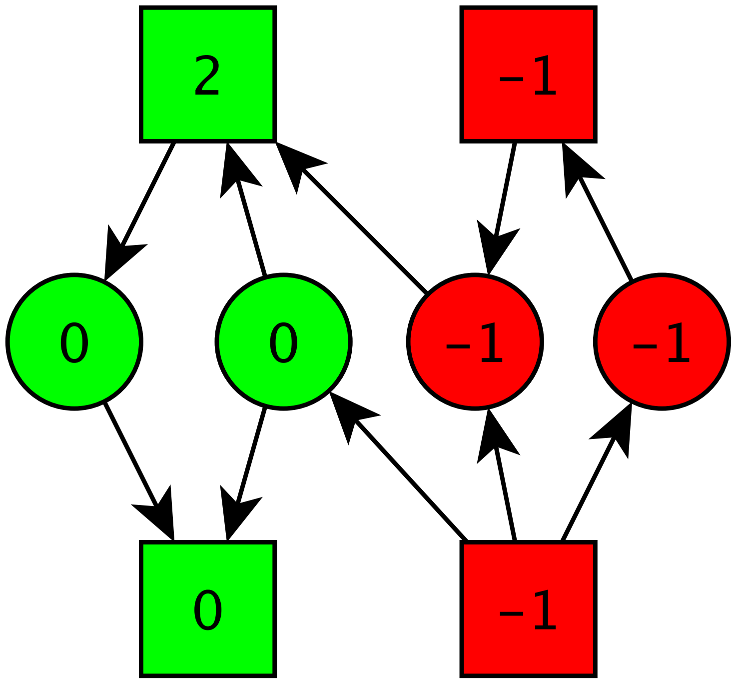


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

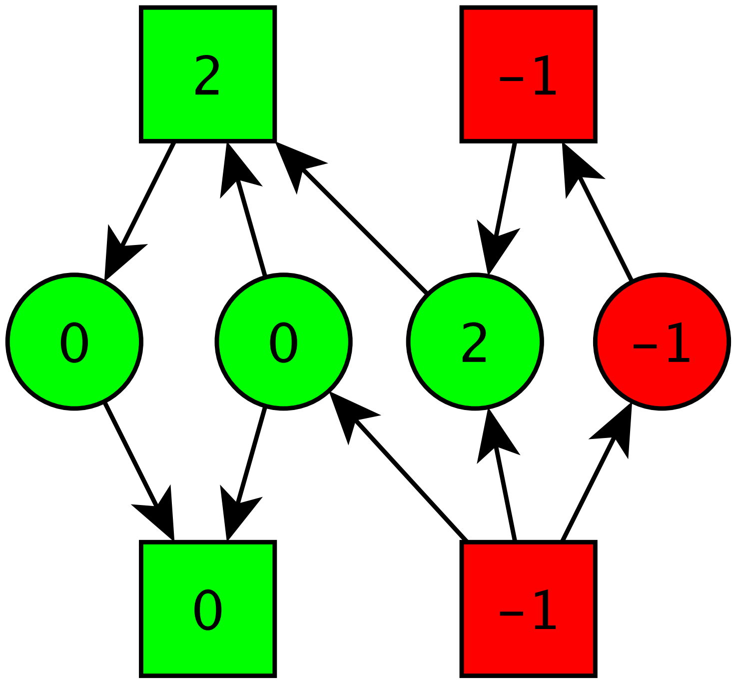


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

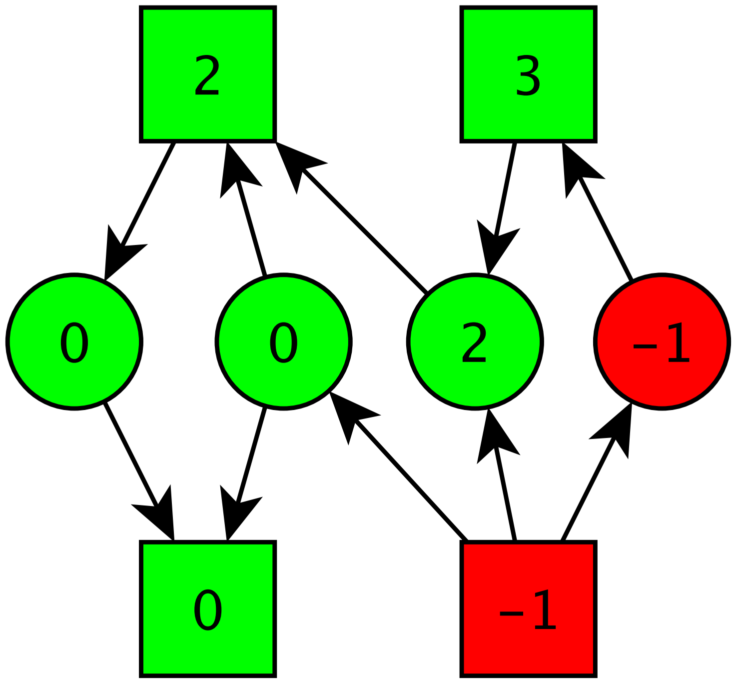


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

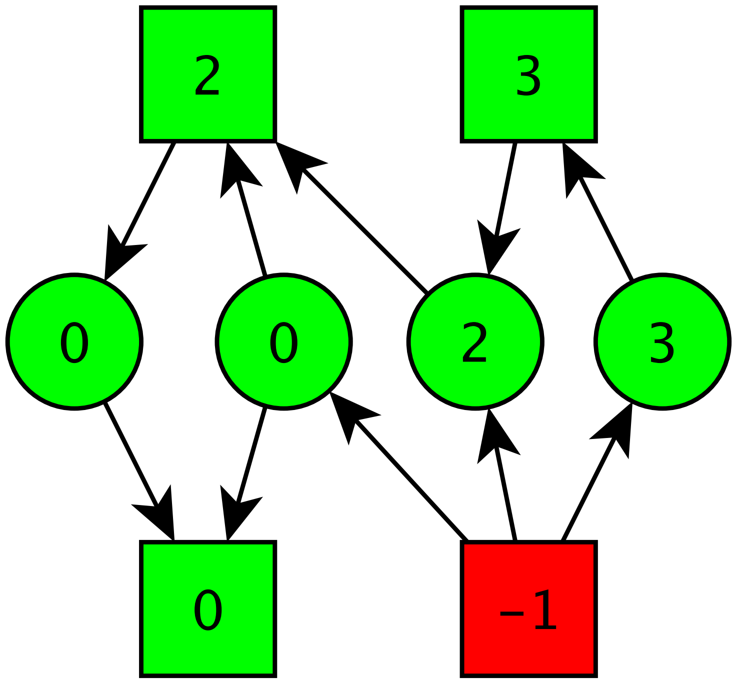


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$

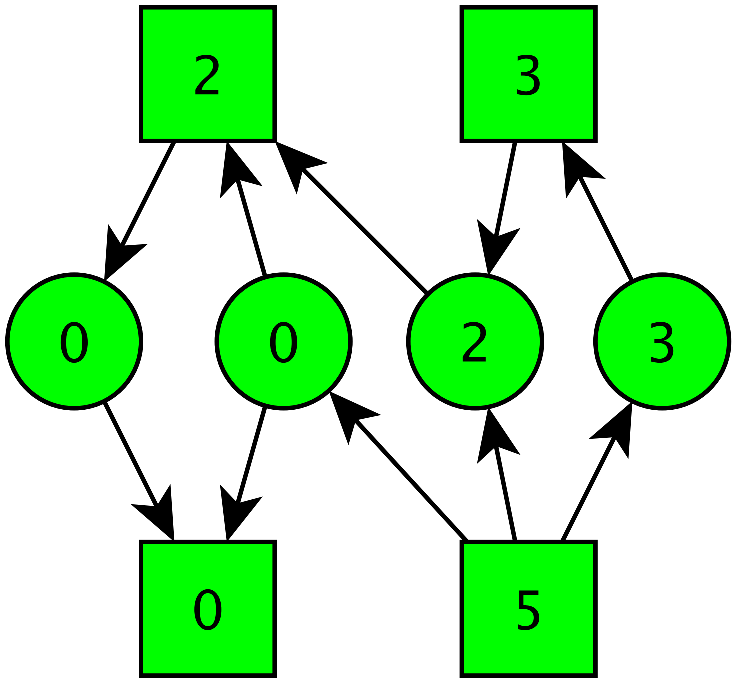


Graph

- $V = \{P_1, P_2, P_3, P_4\}$
- $I = \{P_1, P_2\}$
- $O = \{O_1, O_2\}$
- $\delta = \{P_2, P_3, P_4\}$

$pre(O_1) = \{P_1\}$
$eff(O_1) = \{P_2, P_3\}$
$cost(O_1) = 2$

$pre(O_2) = \{P_3\}$
$eff(O_2) = \{P_4\}$
$cost(O_2) = 1$



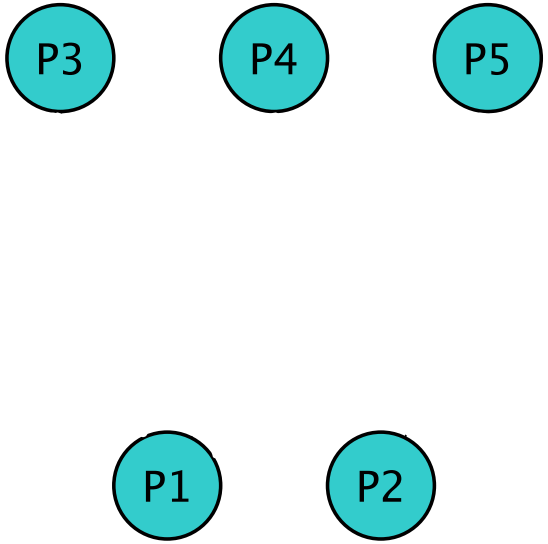
Original Implementation (Fast Downward)

- Operators are split in unary operators
- Easy to construct
- Simple handling of conditional effects

- Many edges

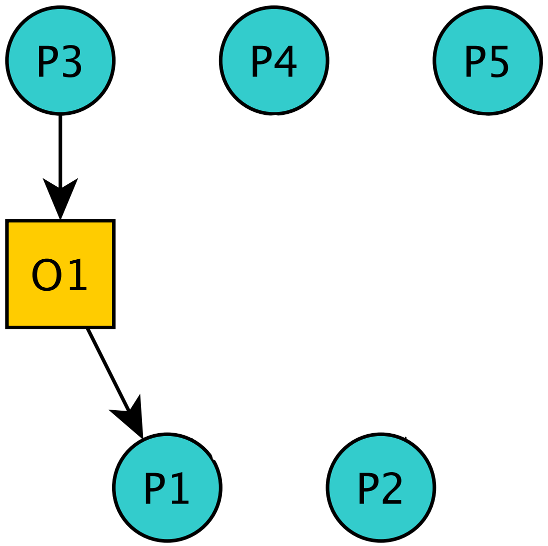
Example

- One operator
- $Precondition\ \{P_1\}$
- $Effect\{P_3, P_4\}$
- $Conditional\ effect\ \{P_5\}$
- $With\ effect\ condition\{P_2\}$



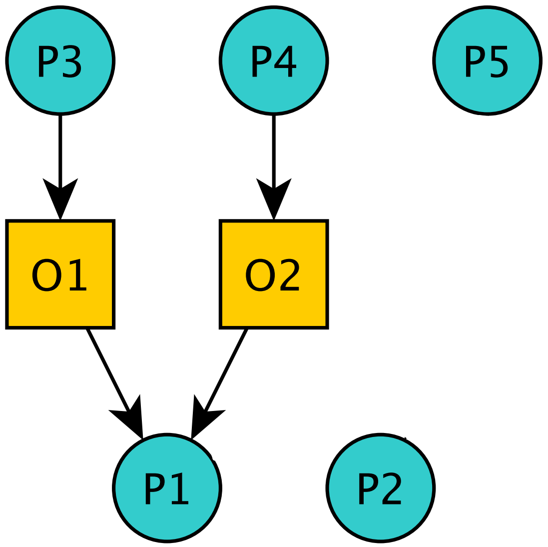
Example

- One operator
- *Precondition* $\{P_1\}$
- *Effect* $\{P_3, P_4\}$
- *Conditional effect* $\{P_5\}$
- *With effect condition* $\{P_2\}$



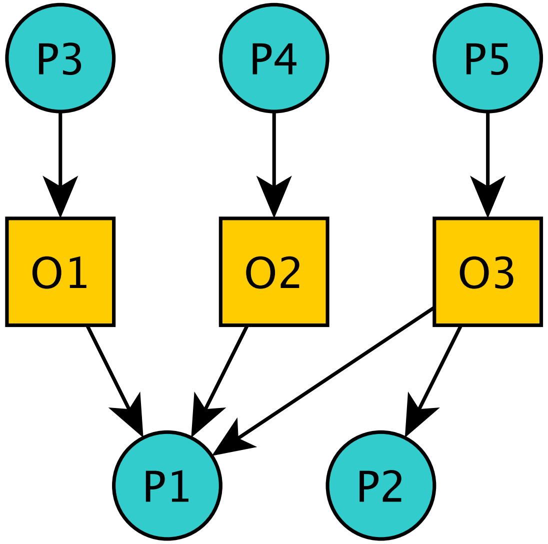
Example

- One operator
- $Precondition\ \{P_1\}$
- $Effect\{P_3, P_4\}$
- $Conditional\ effect\ \{P_5\}$
- $With\ effect\ condition\{P_2\}$



Example

- One operator
- $Precondition\ \{P_1\}$
- $Effect\{P_3, P_4\}$
- $Conditional\ effect\ \{P_5\}$
- $With\ effect\ condition\{P_2\}$

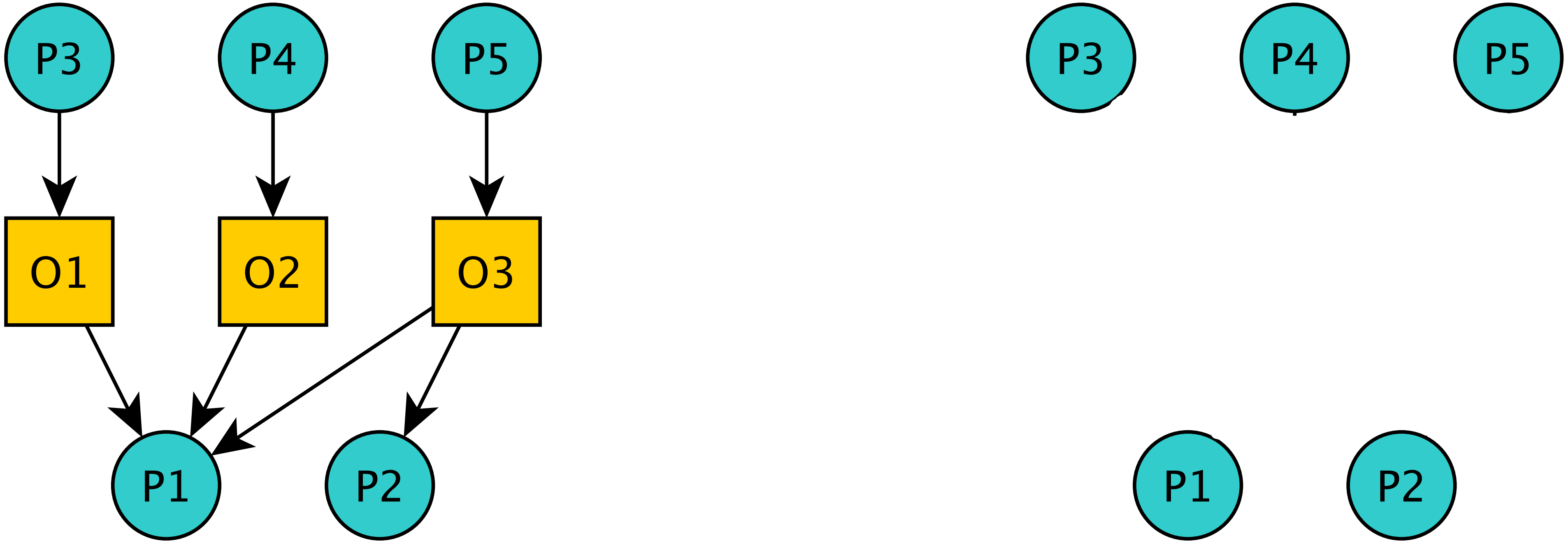


Idea to optimize

- Reduce redundancy

- All effects of an operator in one node
- Conditional effects in separate node

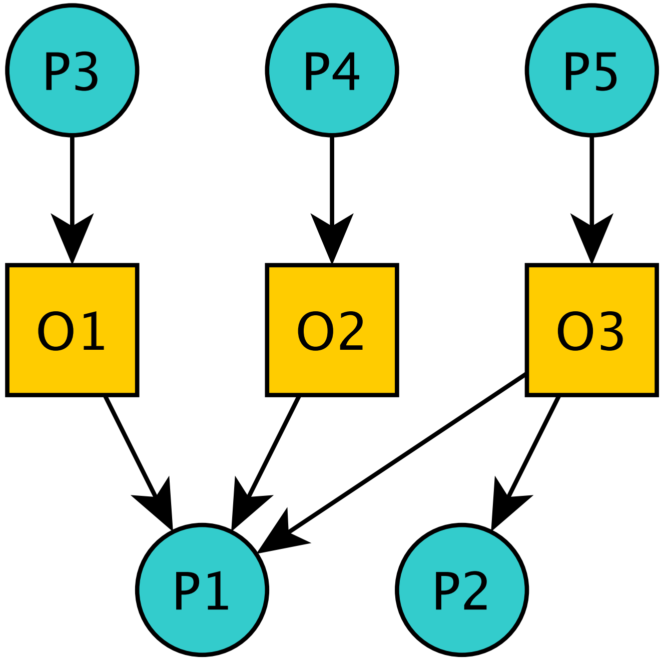
Compare Graphs



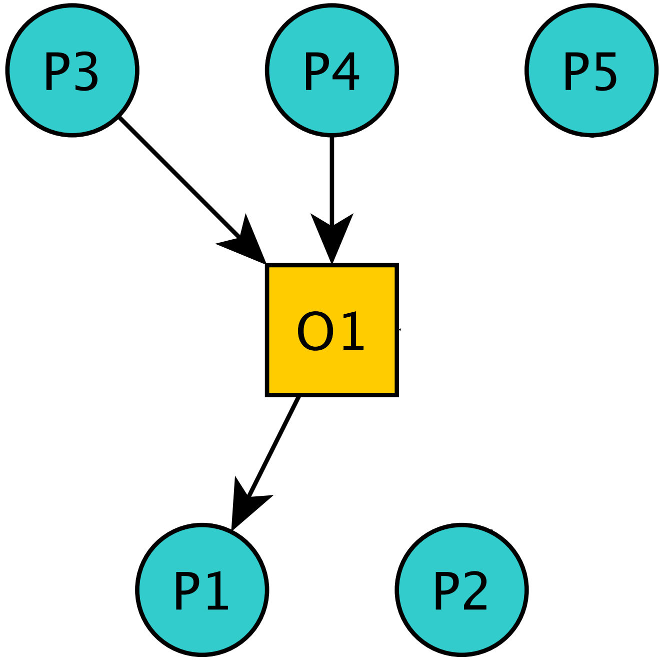
Original Version

New Version

Compare Graphs

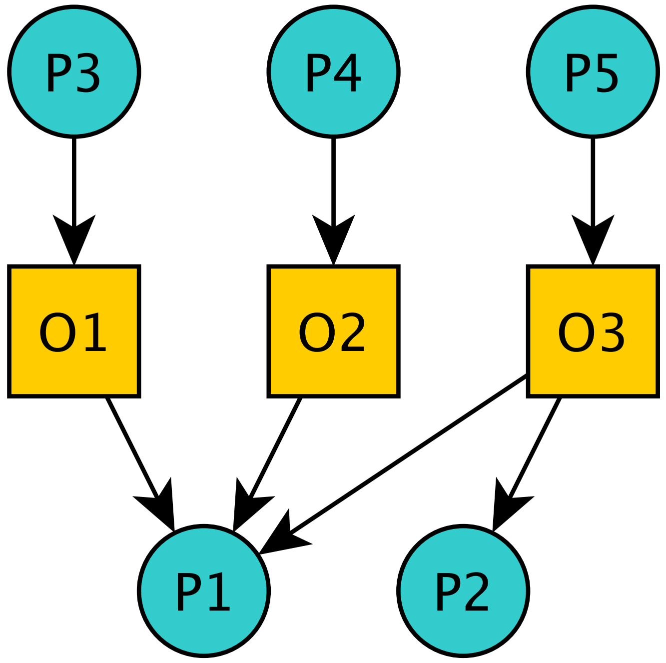


Original Version

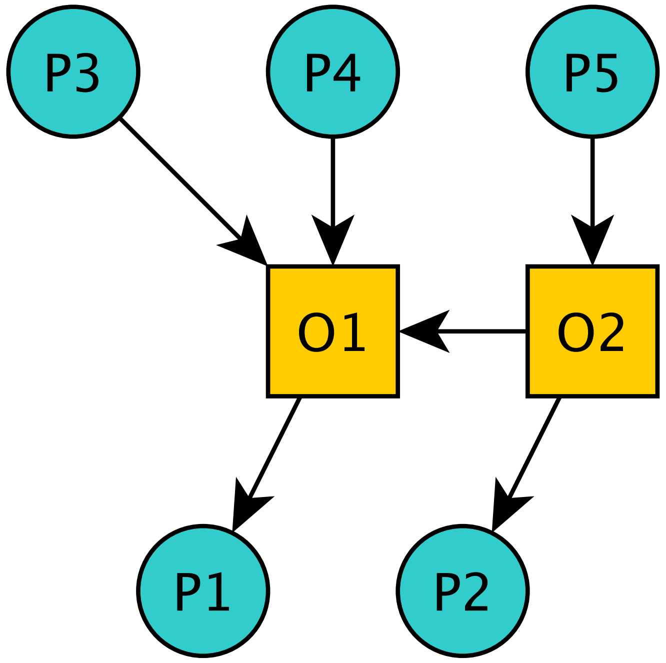


New Version

Compare Graphs



Original Version

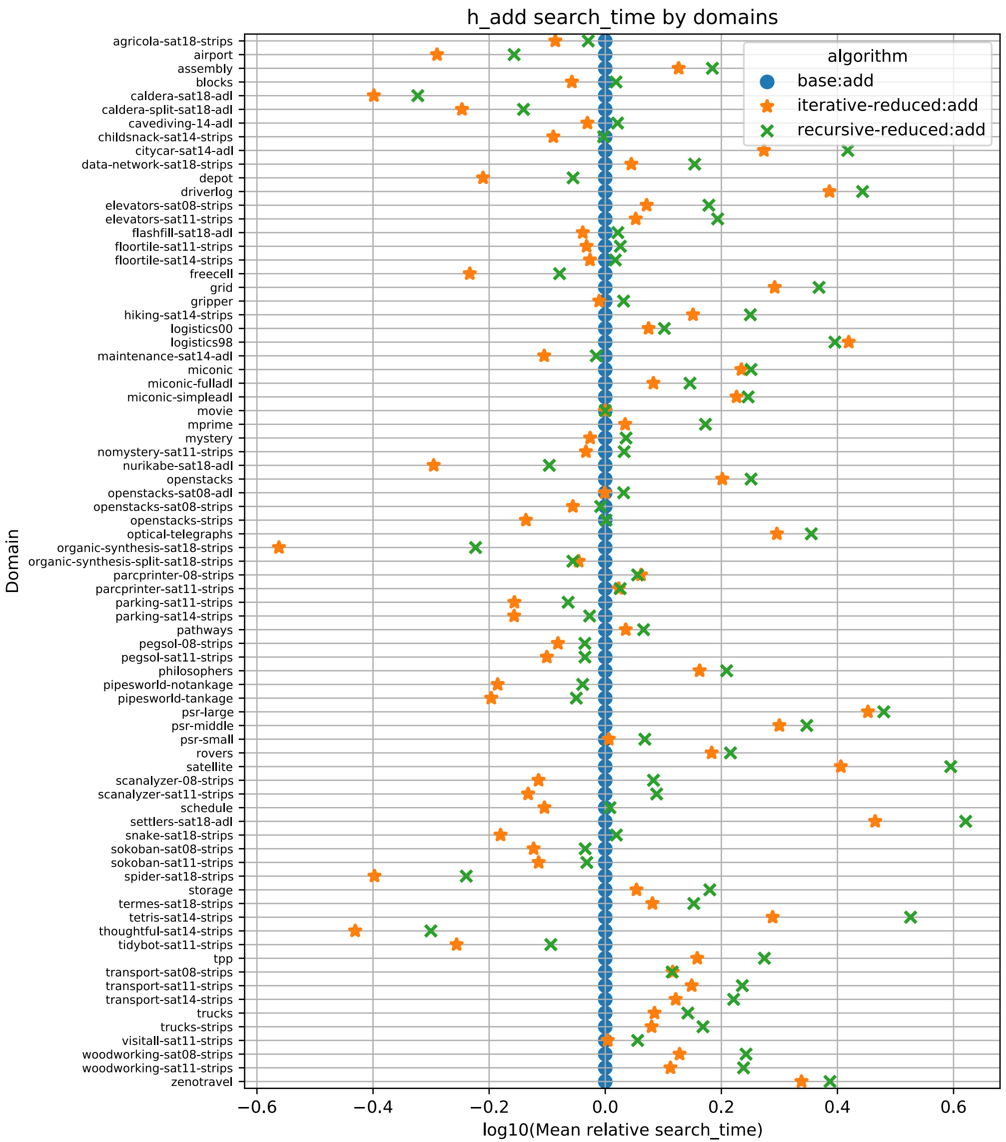


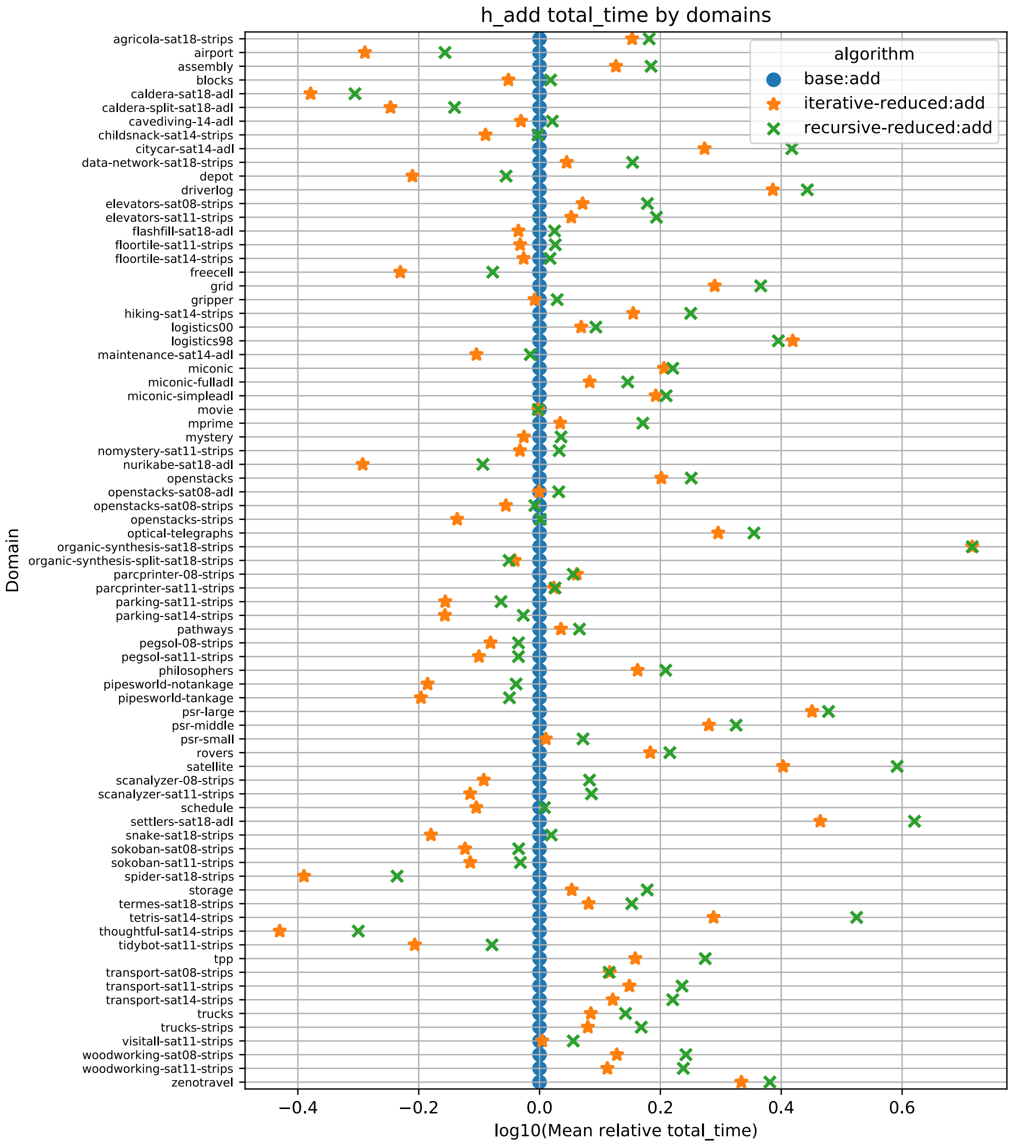
New Version

Results Overall

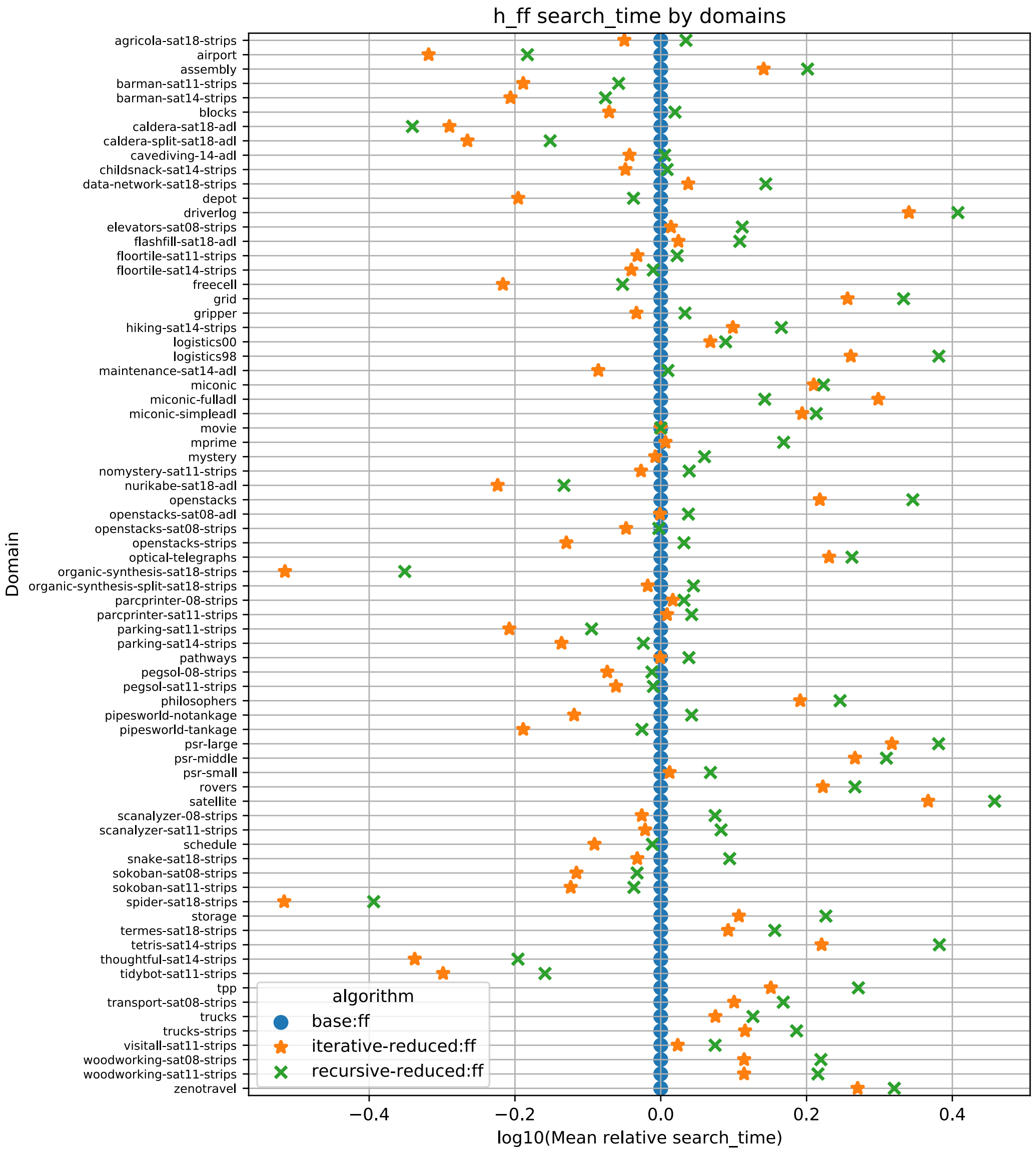
	Implementation	Coverage	R_S	R_T
Additive	Base	1817	1.000	1.000
	Iterative-reduced	1810	1.008	1.057
	Recursive-reduced	1793	1.202	1.237
FF	Base	1760	1.000	1.000
	Iterative-reduced	1771	0.958	1.005
	Recursive-reduced	1754	1.119	1.154
	Out of	2742		

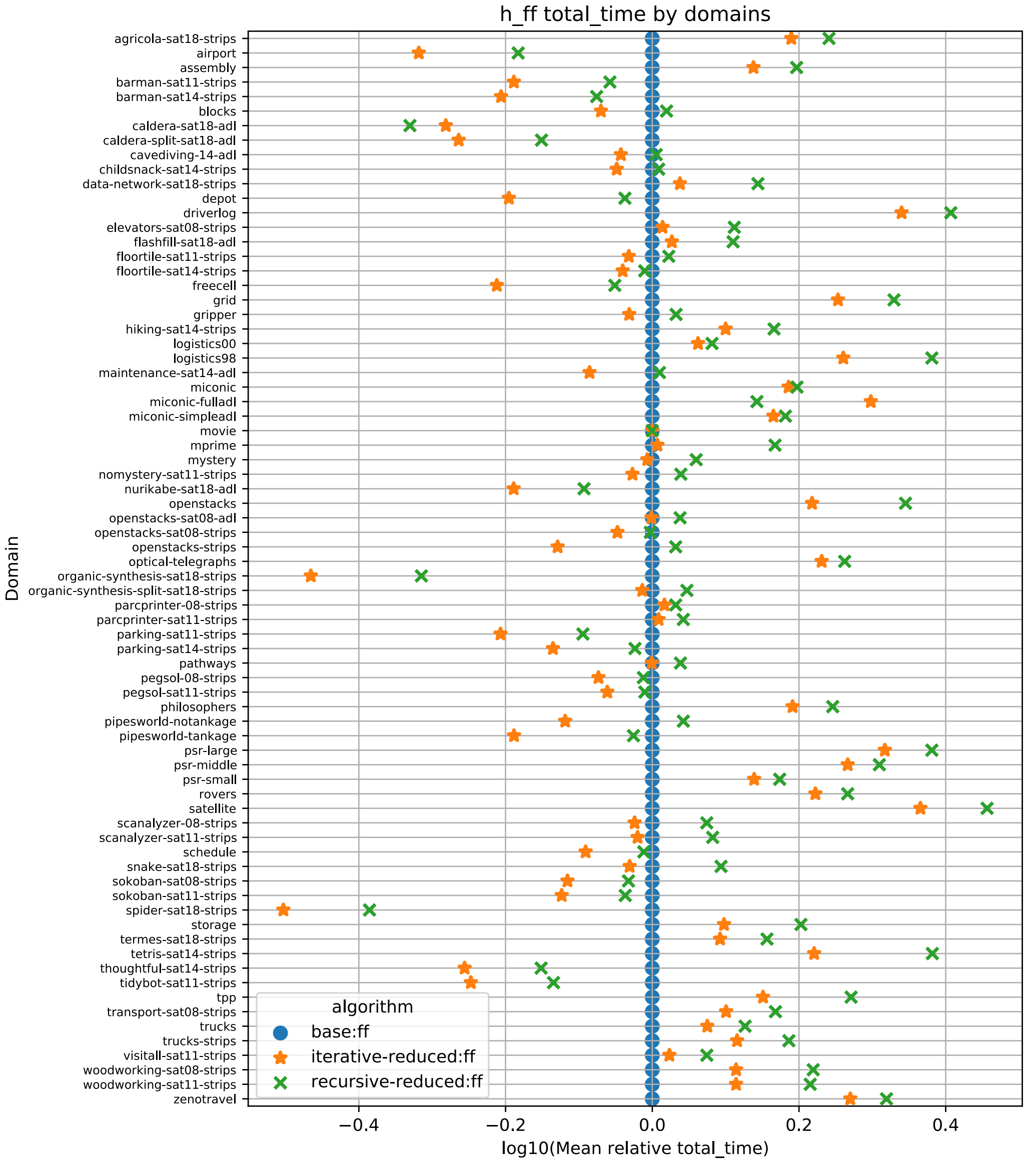
Results by Domain (Additive)





Results by Domain (FF)





Conclusion

- Overall about as fast as the original
- Highly Domain depended

- Original is very well implemented

Future Work

- This thesis focused on operator individually
- Promising to apply over more then one operator