Lifted Successor Generation using Query Optimization Techniques

Augusto B. Corrêa, Florian Pommerening,
Malte Helmert, and Guillem Francès

University of Basel, Switzerland
Universitat Pompeu Fabra, Spain

@ICAPS 2020

Lifted Successor Generation using Query Optimization Techniques

Augusto B. Corrêa, Florian Pommerening,
Malte Helmert, and Guillem Francès

University of Basel, Switzerland
Universitat Pompeu Fabra, Spain

Lifted Successor Generation using Query Optimization Techniques

Augusto B. Corrêa, Florian Pommerening,
Malte Helmert, and Guillem Francès

University of Basel, Switzerland
Universitat Pompeu Fabra, Spain

```
(:action stack
 :parameters (?x ?y)
 :precondition (and (holding ?x)
                    (clear ?y))
 :effect (and (not (holding ?x))
              (not (clear ?y))
              (clear ?x)
              (handempty)
              (on ?x ?y)))
```

```
(:objects block1, block2, ..., block100)
```

```
(:action stack
 :parameters (?x ?y)
 :precondition (and (holding ?x)
                    (clear ?y))
 :effect (and (not (holding ?x))
              (not (clear ?y))
              (clear ?x)
              (handempty)
              (on ?x ?y)))
```

```
(:objects block1, block2, ..., block100)
```

```
(stack block1 block2)
(stack block1 block3)
...
(stack block1 block100)
...
(stack block100 block99)
```

Almost 10.000 ground actions is still fine.

But grounding is not always fine.

Organic Synthesis domain, instance #11:

almost 71.000.000.000.000 ground actions.

Guess the optimal plan length.

2

Grounding is usually fine.

But sometimes it requires 35 trillion times more effort than we need.

What can we do about it?

Lifted Planning: Ground States + Action Schemas

Lifted Planning: Ground States + Action Schemas

```
(:predicates (at ?x ?y) (path ?x ?y))


(:init (at obj1 l1)
       (at obj2 l1)
       (at obj3 l3)
       (at obj4 l2)
       (path l1 l2)
       (path l1 l3)
       (path l2 l3)
       (path l3 l4))
```

at	
obj1	l1
obj2	l1
obj3	l3
obj4	l2

path	
l1	l2
l1	l3
l2	l3
l3	l4

Lifted Planning: Ground States + Action Schemas

Lifted Planning: Ground States + Action Schemas

```
(:precondition
(and (at ?X ?Y)
     (path ?Y ?W)
     (path ?W ?Z)))
```

```
(:precondition
(and (at ?X ?Y)
     (path ?Y ?W)
     (path ?W ?Z)))
```

$$at(X,Y) \bowtie path(Y,W) \bowtie path(W,Z)$$

*These are **conjunctive queries**.*

at	
X	Y
obj1	l1
obj2	l1
obj3	l3
obj4	l2

$\bowtie$

path	
Y	W
l1	l2
l1	l3
l2	l3
l3	l4

$\bowtie$

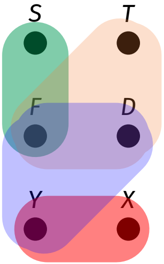
path	
W	Z
l1	l2
l1	l3
l2	l3
l3	l4

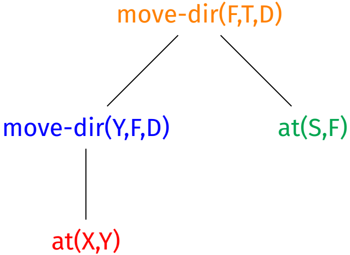
$at(X,Y) \bowtie path(Y,W) \bowtie path(W,Z)$			
X	Y	W	Z
obj1	l1	l2	l3
obj1	l1	l3	l4
obj2	l1	l2	l3
obj2	l1	l3	l4
obj4	l2	l3	l4

Conjunctive queries are NP-hard.

But there's a significant island of tractability.

- 🟥 at(X, Y)
- 🟩 at(S, F)
- 🟦 move-dir(Y, F, D)
- 🟧 move-dir(F, T, D)





Conjunctive queries with join-trees have ***acyclic hypergraphs****.*

They are solvable in output-polynomial time.

Almost 87% of the action schemas in IPC have preconditions with acyclic hypergraphs.

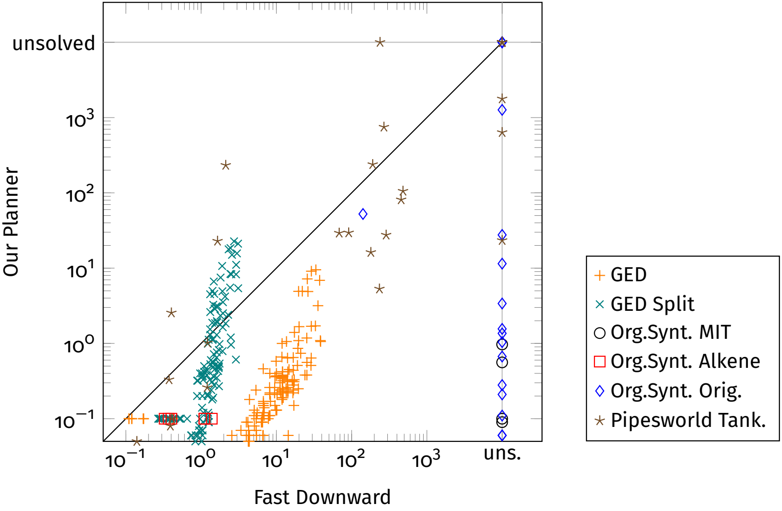
If we focus on hard-to-ground domains, then it is only 21%.

Great part of this is due to inequality constraints.

Processing inequalities:
80% in hard-to-ground domains.

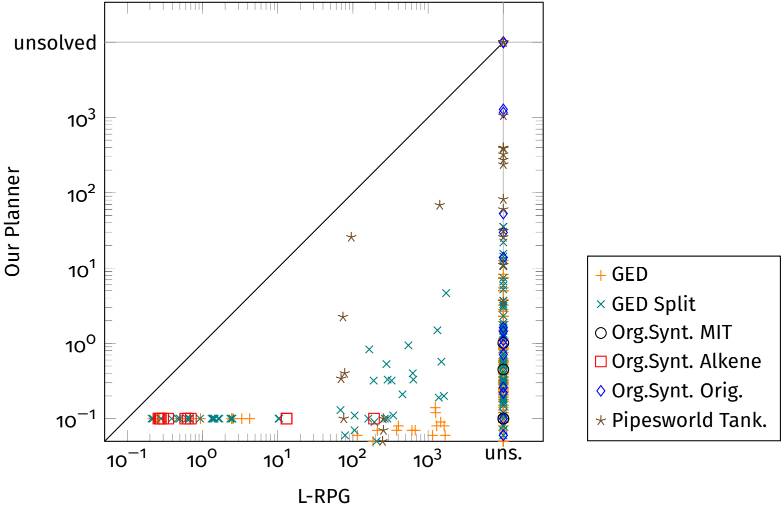
Is this good in practice?

Time (s)



unsolved
10^3
10^2
10^1
10^0
10^{-1}
Our Planner
10^{-1}
10^0
10^1
10^2
10^3
uns.
Fast Downward
+ GED
× GED Split
○ Org.Synt. MIT
□ Org.Synt. Alkene
◇ Org.Synt. Orig.
⋆ Pipesworld Tank.

Time (s)



unsolved
10^3
10^2
10^1
10^0
10^{-1}
Our Planner
10^{-1}
10^0
10^1
10^2
10^3
uns.
L-RPG
+ GED
× GED Split
○ Org.Synt. MIT
□ Org.Synt. Alkene
◇ Org.Synt. Orig.
⋆ Pipesworld Tank.

Conclusions

- Lifted planning can help in hard-to-ground domains.
- Most planning action schemas have acyclic preconditions.
- Much faster than previous state-of-the-art lifted planners.