Conformant Planning as a Benchmark for QBF-Solvers

Martin Kronegger

Joint work with Andreas Pfandler and Reinhard Pichler

Institute of Information Systems
Vienna University of Technology

July 9, 2013
Dungeon Benchmark

Link
Dungeon Benchmark

Pool 1

Pool 2

Link

2 / 7
Dungeon Benchmark

Pool 1

Pool 2

Link

Witch

Parameters

Number of items, special items, unknowns, pools, monsters, etc.

Min./max. size of pools, recipes, requirements for monsters, etc.

Probability of variables to be negated (items, special items, unknowns)
Dungeon Benchmark

Pool 1

Pool 2

Link

Witch

Dungeon

Number of items, special items, unknowns, pools, monsters, etc.
Min./max. size of pools, recipes, requirements for monsters, etc.
Probability of variables to be negated (items, special items, unknowns)
Dungeon Benchmark

Pool 1

Pool 2

Witch

Link

Dungeon
Dungeon Benchmark

Pool 1

Pool 2

Witch

Dungeon

Link
Dungeon Benchmark

Parameters

- Number of items, special items, unknowns, pools, monsters, etc.
- Min./max. size of pools, recipes, requirements for monsters, etc.
- Probability of variables to be negated (items, special items, unknowns)
Architecture of the Solver

PDDL instance → Parser → Grounder → QBF-Encoder → QBF-Solver → Plan

dyn. grounding

No → Yes
Architecture of the Solver

```
(:init (inPool i1 pool1)
  (inPool i4 pool1)
  (inPool i2 pool2)
  (unknown (in u1)) ...
)
(:goal (win g) )
(:action pick
  :parameters (?i - item ?p - pool)
  :effect (and (in ?i) (not (allowPick ?p)))
)  
(:action fight ...
  :effect (and
    (when (and (in i2) (not (in u3)) (in s1)) (win g))
    (when (and (not (in i4)) (in u1)) (win g))
  )
```

PDDL instance → Parser → Grounder → QBF-Encoder → QBF-Solver → Plan
Architecture of the Solver

(:action pick
 :precondition (and (allowPick pool1) (inPool i1 pool1))
 :effect (and (in i1) (not (allowPick pool1)))
)
(:action pick
 :precondition (and (allowPick pool2) (inPool i1 pool2))
 :effect (and (in i1) (not (allowPick pool2)))
)
...
(:action pick
 :precondition (and (allowPick pool1) (inPool i2 pool1))
 :effect (and (in i2) (not (allowPick pool1)))
)
...
Architecture of the Solver

```
(:action pick
  :precondition (and (allowPick pool1) (inPool i1 pool1))
  :effect (and (in i1) (not (allowPick pool1)))
)

(:action fight
  :effect (and
    (when (and (in i2) (not (in i4)) (in s1)) (win g))
    (when (and (not (in i1)) (in u1)) (win g))
  )
)
```

PDDL instance → Parser → Grounder → QBF-Encoder → QBF-Solver

dyn. grounding

Plan
Architecture of the Solver

```
p cnf 1666 4833
e 516 518 512 514 515 1024 1025 1026 1028 1536 ... 0
a 34 118 101 130 46 ... 0
e 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 ... 0
1 0
...
-30 0
...
-139 140 0
-139 -141 0
-177 46 34 130 -118 178 0
-101 193 0
-348 189 190 191 -192 349 0
-348 152 -193 349 0
...
```

```
<table>
<thead>
<tr>
<th>PDDL instance</th>
<th>Parser</th>
<th>Grounder</th>
<th>QBF-Encoder</th>
<th>QBF-Solver</th>
<th>Plan</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
```

dyn. grounding
Encoding without Optimization

based on Rintanen [2007]

Goal: A quantified Boolean formula $\psi$ that is true iff the corresponding planning problem has a plan of length $k$. 

\[
\begin{align*}
\psi &= \exists \text{actions} \forall \text{unknowns} \exists \text{helpers} \\
\phi &= \bigwedge_{v \in \text{PI}} v_0 \land \bigwedge_{v \in \text{NI}} \neg v_0 \land \text{initial state} \land \bigwedge_{v \in \text{PG}} v_k \land \bigwedge_{v \in \text{NG}} \neg v_k \land \text{goal} \land \bigwedge_{t \in [k]} \bigwedge_{a \in A} t-1 \left[ a_{t-1} \land \bigwedge_{p \in \text{pre}(a)} p_{t-1} \right] \supset \bigwedge_{e \in \text{eff}(a)} e_t \land \text{one action at a time} \land \text{every change has a cause (framing axioms)}
\end{align*}
\]
Encoding without Optimization

based on Rintanen [2007]

Goal: A quantified Boolean formula $\psi$ that is true iff the corresponding planning problem has a plan of length $k$.

$$\psi := \exists \text{ actions } \forall \text{ unknowns } \exists \text{ helpers } \varphi$$
Encoding without Optimization

based on Rintanen [2007]

Goal: A quantified Boolean formula \( \psi \) that is true iff the corresponding planning problem has a plan of length \( k \).

\[
\psi : = \exists \text{ actions} \ \forall \text{ unknowns} \ \exists \text{ helpers} \ \varphi \\
\varphi : = \bigwedge_{v \in PI} v^0 \land \bigwedge_{v \in NI} \neg v^0 \land \bigwedge_{v \in PG} v^k \land \bigwedge_{v \in NG} \neg v^k
\]

- initial state
- goal
Encoding without Optimization

based on Rintanen [2007]

Goal: A quantified Boolean formula $\psi$ that is true iff the corresponding planning problem has a plan of length $k$.

$$\psi := \exists \text{ actions } \forall \text{ unknowns } \exists \text{ helpers } \varphi$$

$$\varphi := \bigwedge_{v \in PI} v^0 \land \bigwedge_{v \in NI} \neg v^0 \land \bigwedge_{v \in PG} v^k \land \bigwedge_{v \in NG} \neg v^k$$

initial state

goal

$$\land \bigwedge_{t \in [k]} \bigwedge_{a \in A^{t-1}} \left[ \left( a^{t-1} \land \bigwedge_{p \in \text{pre}(a)} p^{t-1} \right) \supset \bigwedge_{e \in \text{eff}(a)} e^t \right]$$
Encoding without Optimization

based on Rintanen [2007]

Goal: A quantified Boolean formula $\psi$ that is true iff the corresponding planning problem has a plan of length $k$.

$$\psi := \exists \text{ actions } \forall \text{ unknowns } \exists \text{ helpers } \varphi$$

$$\varphi := \bigwedge_{v \in PI} v^0 \land \bigwedge_{v \in NI} \neg v^0 \land \bigwedge_{v \in PG} v^k \land \bigwedge_{v \in NG} \neg v^k$$

$\bigwedge_{t \in [k]} \bigwedge_{a \in A^{t-1}} \left[ \left( a^{t-1} \land \bigwedge_{p \in \text{pre}(a)} p^{t-1} \right) \supset \bigwedge_{e \in \text{eff}(a)} e^t \right]$ 

$\land \text{ one action at a time}$
Encoding without Optimization

based on Rintanen [2007]

Goal: A quantified Boolean formula $\psi$ that is true iff the corresponding planning problem has a plan of length $k$.

$$\psi := \exists \text{ actions } \forall \text{ unknowns } \exists \text{ helpers } \varphi$$

$$\varphi := \bigwedge_{v \in PI} v^0 \land \bigwedge_{v \in NI} \neg v^0 \land \bigwedge_{v \in PG} v^k \land \bigwedge_{v \in NG} \neg v^k$$

- $\bigwedge_{v \in PI} v^0$ initial state
- $\bigwedge_{v \in NI} \neg v^0$ goal
- $\bigwedge_{v \in PG} v^k$ initial state
- $\bigwedge_{v \in NG} \neg v^k$ goal

$$\bigwedge_{t \in [k]} \bigwedge_{a \in A^{t-1}} \left[ \left( a^{t-1} \land \bigwedge_{p \in \text{pre}(a)} p^{t-1} \right) \supset \bigwedge_{e \in \text{eff}(a)} e^t \right]$$

- $\bigwedge_{t \in [k]} \bigwedge_{a \in A^{t-1}}$ one action at a time
- $\bigwedge_{p \in \text{pre}(a)}$ every change has a cause (framing axioms)
Advantages of this System and Benchmark Environment

Main advantages

- Exact solver (if plan is found, then the plan has minimal length)
- Conformant planning benchmarks accessible to QBF-solvers

Benchmark environment

- Randomly generated Dungeon instances
- Different QBF-solvers (AQME, DepQBF, QuBE, sKizzo)
- Compare to other planning tools (ConformantFF, T0)
Benchmark Results

![Graph showing benchmark results]

- AQME
- ConfFF
- DepQBF
- QuBE
- sKizzo
- T0

**Plan Length vs. Instance Number**

- Number of instances solved out of 144.
### Benchmark Results

<table>
<thead>
<tr>
<th>#</th>
<th>#i</th>
<th>#m</th>
<th>#u</th>
<th>AQME t/pl</th>
<th>ConfFF t/pl</th>
<th>DepQBF t/pl</th>
<th>QuBE t/pl</th>
<th>sKizzo t/pl</th>
<th>T0 t/pl</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>10</td>
<td>5</td>
<td>10</td>
<td>603.53</td>
<td>0.04 unsol</td>
<td>27478</td>
<td>9031</td>
<td>11826</td>
<td>0.43 unsol?</td>
</tr>
<tr>
<td>40</td>
<td>10</td>
<td>20</td>
<td>4</td>
<td>2.24</td>
<td>0.25</td>
<td>0.29</td>
<td>0.32</td>
<td>1733</td>
<td>0.09</td>
</tr>
<tr>
<td>72</td>
<td>15</td>
<td>75</td>
<td>10</td>
<td>428.26</td>
<td>1.06</td>
<td>0.92</td>
<td>-</td>
<td>5398</td>
<td>565.85</td>
</tr>
<tr>
<td>143</td>
<td>30</td>
<td>150</td>
<td>5</td>
<td>690.02</td>
<td>4</td>
<td>5</td>
<td>-</td>
<td>48988</td>
<td>to</td>
</tr>
</tbody>
</table>

| solved instances (out of 144) | 96 | 64 | 80 | 81 | 76 | 102 |

The table above shows the benchmark results for different instances, with columns indicating the number of instances (#), instance number (#i), number of models (#m), number of unique models (#u), and various metrics for each solver (AQME, ConfFF, DepQBF, QuBE, sKizzo, T0). The metrics include time per plan (t/pl) and the number of clauses and variables (cl/vars). The results are highlighted in green where the solver finds a solution within the time limit, or red where it times out.
Conclusion

- System for planning with uncertainty in the initial state
- Conformant planning benchmarks are accessible to QBF-solvers
- Approach sometimes competitive to state-of-the-art planners
Conclusion

- System for planning with uncertainty in the initial state
- Conformant planning benchmarks are accessible to QBF-solvers
- Approach sometimes competitive to state-of-the-art planners

Wishlist for further improvements

- Certificates (or at least assignment to outermost quantifier-block)
- Incremental QBF-solvers
- QBF-solvers biased towards unsatisfiability
Conclusion

- System for planning with uncertainty in the initial state
- Conformant planning benchmarks are accessible to QBF-solvers
- Approach sometimes competitive to state-of-the-art planners

Wishlist for further improvements

- Certificates (or at least assignment to outermost quantifier-block)
- Incremental QBF-solvers
- QBF-solvers biased towards unsatisfiability

Future work

- More detailed experiments
- Experiments with other planning domains
- Solver specific encodings
- Improving dynamic grounding