Certifying Planning Systems: Witnesses for Unsolvability

Salomé Eriksson

University of Basel, Switzerland

April 26, 2019
Classical Planning
Validating Planner Output

- **Why?**
  - software bugs
  - hardware faults
  - malicious reasons
  - ...

- **How?**
  - tests on known instances
  - formal correctness proofs
  - certifying algorithms
Certifying Algorithms

generate a *witness* alongside answer:

![Diagram of certifying algorithms]

- **Task**: Input
  - **Plan**: Output
  - **Planner**: "solvable"
  - **Plan validation tool**: "valid"/"invalid"
Certifying Algorithms

generate a **witness** alongside answer:
**Main Contributions**

- two suitable witness types for unsolvable planning tasks:
  - Inductive Certificates
  - Proof System

- theoretical and experimental comparison

**suitability measures:**

- soundness & completeness
- efficient generation and verification
- generality
Witness I: Inductive Certificates

[E, Röger, Helmert, ICAPS 2017]
Inductive Sets

Inductive Sets can only reach states with "box in corner".
Inductive Sets

Inductive Sets can only reach states with "box in corner".
Inductive Sets

Inductive Sets can only reach states with "box in corner."
Inductive Sets

Inductive Sets can only reach states with "box in corner".
Inductive Sets
Inductive Sets

A set of states is **inductive** if all action applications to a state in $S$ lead to a state which is also in $S$. ($S[A] \subseteq S$).

can only reach states with “box in corner”
Inductive Certificate

set of states $S$ with following properties:

- contains $I$
- contains no goal
- inductive
Theorem

Inductive certificates are sound and complete.

states reachable from $I$:

- contains $I$
- is inductive
- contains no goal if task solvable
Efficient Verification

depends on how $S$ is represented

- formalisms based on propositional logic
- Which logical operations are needed for efficient verification?

several commonly used formalisms support needed operations
Composite Certificates

not all sets can be compactely described
⇒ represent as union or intersection of sets

$r$-disjunctive Certificates

family $\mathcal{F}$ of sets with:

- $I \in S$ for some $S \in \mathcal{F}$
- no goal in any $S \in \mathcal{F}$
- $S[a] \subseteq \bigcup_{S' \in \mathcal{F}'} S'$ for all $a \in A$, $S \in \mathcal{F}$ with $\mathcal{F}' \subseteq \mathcal{F}$ and $|\mathcal{F}'| \leq r$. 
Application to Heuristic Search

**heuristic** can detect dead-ends

\[ \leadsto \text{set of reachable states not explored fully} \]
**Application to Heuristic Search**

- **walk-up**
- **walk-right**
- **push-right**
- **push-up**

$h = \infty$
heuristic can detect dead-ends
\[\leadsto\] set of reachable states not explored fully

Heuristic Search Certificate

Union of:
- inductive set for each dead-end
  - for each \( a \in A \): leads to itself
Application to Heuristic Search

walk-up

walk-right

push-right

push-up

$\ h = \infty \ $

$\ h = \infty \ $
Application to Heuristic Search

heuristic can detect dead-ends
\[ \leadsto \text{set of reachable states not explored fully} \]

Heuristic Search Certificate

Union of:
- inductive set for each dead-end
  - for each \( a \in A \): leads to itself
- one set for each expanded state
  - for each \( a \in A \): leads to one expanded or dead-end state

\[ \leadsto 1\text{-disjunctive} \]
Application to Heuristic Search

walk-up

walk-right

push-right

$h = \infty$

push-up

$h = \infty$
Generating Inductive Certificates

| certificates      |  
|-------------------|---
| blind search      | yes |
| heuristic search  |  
| - single heuristic| yes |
| - several heuristics| **if same formalism** |
| $h^+$              | yes |
| $h^m$             | yes |
| $h^{M&S}$         | yes |
| Landmarks         | yes |
| Trapper           | yes |
| Iterative dead pairs |  
| CLS               | yes |
Weaknesses

**monolithic**: find one inductive set

- cannot mix representations
  - several heuristics
- cannot cover techniques not built on inductive sets
  - iterative dead pairs
Witness II: Proof System

[E, Röger, Helmert, ICAPS 2018]
Dead States

incrementally rule out parts of the search space

**Definition**

A state $s$ is dead if no plan traverses $s$.
A set of states is dead if all its elements are dead.

initial state / all goal states dead $\rightarrow$ task unsolvable
Proof Systems

based on rules with premises $A_i$ and conclusion $B$:

$$
\begin{array}{c}
A_1 \\
\vdots \\
A_n \\
\hline \\
B
\end{array}
$$

universally true
Rules

- showing that state sets are dead
- end proof
- set theory
showing that state sets are dead

end proof

set theory
• showing that state sets are dead
• end proof
• set theory
Rules

- showing that state sets are dead
- end proof
- set theory

\[
\begin{align*}
I \text{ dead} & \quad \text{unsolvable} \\
G \text{ dead} & \quad \text{unsolvable}
\end{align*}
\]
Rules

- showing that state sets are dead
- end proof
- set theory

\[
S \subseteq (S \cup S')
\]

\[
\begin{align*}
S & \subseteq S' & S' & \subseteq S'' \\
\hline
S & \subseteq S''
\end{align*}
\]
Basic Statements

show $S \subseteq S'$ holds for concrete sets?
$
\rightsquigarrow \text{basic statements}$

- verified for concrete task
- establish "initial" knowledge base
Soundness & Completeness

Theorem

Proofs in the proof system are sound and complete.

inductive certificate $S$:

- no successor
- containing $I$
- no goal

(1) $\emptyset$ dead
(2) $S[A] \subseteq S \cup \emptyset$
(3) $S \cap G \subseteq \emptyset$
(4) $S \cap G$ dead
(5) $S$ dead
(6) $I \in S$
(7) $I$ dead
(8) unsolvable
Efficient Verification

rule verification trivial \(\leadsto\) only depends on basic statements

different forms of \(S \subseteq S'\):

- \(S'\) as a intersection of sets
- \(S'\) as a union of sets
- \(S\) and \(S'\) represented in different formalisms

translated inductive certificates require same operations
Heuristic Search Proof

proof structure:

1. each dead end is dead (inductive set)
Application to Heuristic Search

$\text{walk-up}$

$\text{walk-right}$

$\text{push-right}$

$\text{push-up}$

$h = \infty$
Application to Heuristic Search

Heuristic Search Proof

proof structure:

1. each dead end is dead (inductive set)
2. union of all dead ends is dead
Application to Heuristic Search

$\mathbf{h} = \infty$

walk-up

walk-right

push-right

push-up

$h = \infty$

...
Application to Heuristic Search

Heuristic Search Proof

proof structure:

1. each dead end is dead (inductive set)
2. union of all dead ends is dead
3. \( \text{expanded}[A] = \text{expanded} \cup \text{dead} \leadsto \text{expanded dead} \)
4. \( I \in \text{expanded} \leadsto I \text{ dead}. \)
Application to Heuristic Search

- walk-up
- walk-right
- push-right
- push-up

$h = \infty$

...
# Generating Proofs

<table>
<thead>
<tr>
<th>Method</th>
<th>Certificates</th>
<th>Proofs</th>
</tr>
</thead>
<tbody>
<tr>
<td>blind search</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>heuristic search</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- single heuristic</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>- several heuristics</td>
<td>if same formalism</td>
<td>yes</td>
</tr>
<tr>
<td>$h^+$</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>$h^m$</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>$h^{M&amp;S}$</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Landmarks</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Trapper</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Iterative dead pairs</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>CLS</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
Comparison
### Theoretical Comparison

- both witnesses sound & complete
- proof covers more examined techniques
- translation certificate $\rightarrow$ proof possible
  - also for composite certificates, but at cost of size increase

$\implies$ proof system more expressive
Experimental Evaluation

comparison for A* search with

- $h^{\text{max}}$
- $h^{\text{M&S}}$

limits:

- generate: 30 minutes
- verify: 4 hours
Coverage - Generation

$h^{\text{max}}$

- Certificate: 175
- Proof: 172

$h^{\text{M&S}}$

- Certificate: 207
- Proof: 212

225

242
Coverage - Verification

$h_{\text{max}}$

- Certificate: 146
- Proof: 156

$h_{\text{M&S}}$

- Certificate: 187
- Proof: 198
Verification

certificate repeats explicit search
Witness Size

- Witness I: Inductive Certificates
- Witness II: Proof System
- Comparison
- Conclusion

Certificate size (in MiB) vs. proof size (in MiB)

- $h_{\text{max}}$
- $h_{\text{M&S}}$

Failed cases represented in the graph.
Conclusion
Summary

Inductive Certificates

- describes invariant property which $I$ has but not $G$
- concise argument for unsolvability
- lacks composability

Proof System

- explicit reasoning with simple rules
- versatile and extensible
<table>
<thead>
<tr>
<th></th>
<th>BDD</th>
<th>Horn</th>
<th>2CNF</th>
<th>MODS</th>
</tr>
</thead>
<tbody>
<tr>
<td>MO</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>CO</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>VA</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>CE</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>IM</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>SE</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>ME</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>∧BC</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>∧C</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>∨BC</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no*</td>
</tr>
<tr>
<td>∨C</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>¬C</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>CL</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>RN</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>RN≺</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>toDNF</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>toCNF</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>CT</td>
<td>yes</td>
<td>(no)</td>
<td>(no)</td>
<td>yes</td>
</tr>
</tbody>
</table>
Traditional:

\[ \varphi \land \bigwedge_{v_p \in pre(a)} v_p \land \bigwedge_{v_a \in add(a)} v'_a \land \bigwedge_{v_d \in (del(a) \setminus add(a))} \neg v'_d \land \bigwedge_{v \in (V \Pi \setminus (add(a) \cup del(a)))} (v \leftrightarrow v') \models \varphi[V \rightarrow V'] \]

New:

\[ \left( (\varphi \land \bigwedge_{v_p \in pre(a)} v_p) \left[ (add(a) \cup del(a)) \rightarrow X' \right] \right) \land \bigwedge_{v_a \in add(a)} v_a \land \bigwedge_{v_d \in (del(a) \setminus add(a))} \neg v_d \models \varphi \]
For $r \in \mathbb{N}_0$, a family $\mathcal{F} \subseteq 2^{S^\Pi}$ of state sets of task $\Pi = \langle V^\Pi, A^\Pi, I^\Pi, G^\Pi \rangle$ is called an $r$-disjunctive certificate if:

1. $I^\Pi \in S$ for some $S \in \mathcal{F}$,
2. $S \cap S^\Pi_G = \emptyset$ for all $S \in \mathcal{F}$, and
3. for all $S \in \mathcal{F}$ and all $a \in A^\Pi$, there is a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ with $|\mathcal{F}'| \leq r$ such that $S[a] \subseteq \bigcup_{S' \in \mathcal{F}'} S'$. 

\[ \text{Disjunctive Certificates} \]
Disjunctive Certificates
Conjunctive Certificates

For $r \in \mathbb{N}_0$, a family $\mathcal{F} \subseteq 2^{S^\Pi}$ of state sets of task $
abla = \langle V^\Pi, A^\Pi, I^\Pi, G^\Pi \rangle$ is called an $r$-conjunctive certificate if:

1. $I^\Pi \in S$ for all $S \in \mathcal{F}$,
2. there is a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ with $|\mathcal{F}'| \leq r$ such that $(\bigcap_{S \in \mathcal{F}'} S) \cap S^\Pi G = \emptyset$, and
3. for all $S \in \mathcal{F}$ and all $a \in A^\Pi$, there is a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ with $|\mathcal{F}'| \leq r$
   such that $(\bigcap_{S' \in \mathcal{F}'} S')[a] \subseteq S$. 

$r$-conjunctive certificate
Conjunctive Certificates

\[ S_1 \cap S_2 \cap S_3 \]

\[ a_1 \rightarrow a_2 \]
### Proof System Rules

<table>
<thead>
<tr>
<th>Rule Type</th>
<th>Premise</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Empty set Dead</td>
<td>$\emptyset$ dead</td>
<td>$E$</td>
</tr>
<tr>
<td>Union Dead</td>
<td>$S$ dead $S'$ dead</td>
<td>$S \cup S'$ dead</td>
</tr>
<tr>
<td>Subset Dead</td>
<td>$S'$ dead $S \subseteq S'$</td>
<td>$S$ dead</td>
</tr>
<tr>
<td>Progression Goal</td>
<td>$S[A^\Pi] \subseteq S \cup S'$ $S'$ dead $S \cap S_G^\Pi$ dead</td>
<td>$S$ dead</td>
</tr>
<tr>
<td>Progression Initial</td>
<td>$S[A^\Pi] \subseteq S \cup S'$ $S'$ dead ${I^\Pi} \subseteq S$</td>
<td>$S$ dead</td>
</tr>
<tr>
<td>Regression Goal</td>
<td>$[A^\Pi]S \subseteq S \cup S'$ $S'$ dead $\overline{S} \cap S_G^\Pi$ dead</td>
<td>$\overline{S}$ dead</td>
</tr>
<tr>
<td>Regression Initial</td>
<td>$[A^\Pi]S \subseteq S \cup S'$ $S'$ dead ${I^\Pi} \subseteq \overline{S}$</td>
<td>$S$ dead</td>
</tr>
</tbody>
</table>
Proof System Rules

**Conclusion Initial**

\[
\text{\{I}^{\Pi}\text{\}} \text{ dead} \quad \frac{\text{unsolvable}}{\text{CI}}
\]

**Conclusion Goal**

\[
S_\text{G}^{\Pi} \text{ dead} \quad \frac{\text{unsolvable}}{\text{CG}}
\]
Proof System Rules

**Union Right**

\[ E \subseteq (E \cup E') \]  \( \text{UR} \)

**Union Left**

\[ E \subseteq (E' \cup E) \]  \( \text{UL} \)

**Intersection Right**

\[ (E \cap E') \subseteq E \]  \( \text{IR} \)

**Intersection Left**

\[ (E' \cap E) \subseteq E \]  \( \text{IL} \)

**Distributivity**

\[ ((E \cup E') \cap E'') \subseteq ((E \cap E'') \cup (E' \cap E'')) \]  \( \text{DI} \)

**Subset Union**

\[ E \subseteq E'' \quad E' \subseteq E'' \]

\[ (E \cup E') \subseteq E'' \]  \( \text{SU} \)

**Subset Intersection**

\[ E \subseteq E' \quad E \subseteq E'' \]

\[ E \subseteq (E' \cap E'') \]  \( \text{SI} \)

**Subset Transitivity**

\[ E \subseteq E' \quad E' \subseteq E'' \]

\[ E \subseteq E'' \]  \( \text{ST} \)
### Proof System Rules

**Action Transitivity**

\[
\frac{S[A] \subseteq S' \quad A' \subseteq A}{S[A'] \subseteq S'} \quad \text{AT}
\]

**Action Union**

\[
\frac{S[A] \subseteq S' \quad S[A'] \subseteq S'}{S[A \cup A'] \subseteq S'} \quad \text{AU}
\]

**Progression Transitivity**

\[
\frac{S[A] \subseteq S'' \quad S' \subseteq S}{S'[A] \subseteq S''} \quad \text{PT}
\]

**Progression Union**

\[
\frac{S[A] \subseteq S'' \quad S'[A] \subseteq S''}{(S \cup S')[A] \subseteq S''} \quad \text{PU}
\]

**Progression to Regression**

\[
\frac{S[A] \subseteq S'}{[A \overline{S'}] \subseteq \overline{S}} \quad \text{PR}
\]

**Regression to Progression**

\[
\frac{S[A] \subseteq S'}{S[A] \subseteq S'} \quad \text{RP}
\]
Proof System Basic Statements

1. \( \bigcap_{L_R \in L} L_R \subseteq \bigcup_{L'_R \in L'} L'_R \)
   with \( |L| + |L'| \leq r \)

2. \( (\bigcap_{X_R \in X} X_R)[A] \cap \bigcap_{L_R \in L} L_R \subseteq \bigcup_{L'_R \in L'} L'_R \)
   with \( |X| + |L| + |L'| \leq r \)

3. \( [A](\bigcap_{X_R \in X} X_R) \cap \bigcap_{L_R \in L} L_R \subseteq \bigcup_{L'_R \in L'} L'_R \)
   with \( |X| + |L| + |L'| \leq r \)

4. \( L_R \subseteq L'_R \)

5. \( A \subseteq A' \)
Proof System Basic Statements

\[ \bigcap_{L_i \in \mathcal{L}} L_i \subseteq \bigcup_{L_i' \in \mathcal{L}'} L_i' : \]

<table>
<thead>
<tr>
<th>[\mathcal{L}^- + \mathcal{L}^+ = 0]</th>
<th>[\mathcal{L}^- + \mathcal{L}^+ = 1]</th>
<th>[\mathcal{L}^- + \mathcal{L}^+ &gt; 1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>[\mathcal{L}^- + \mathcal{L}^+ = 0]</td>
<td>[\mathcal{L}^+ + \mathcal{L}^- = 0]</td>
<td>[\mathcal{L}^+ + \mathcal{L}^- = 1]</td>
</tr>
<tr>
<td>[\mathcal{L}^- + \mathcal{L}^+ = 1]</td>
<td>[\mathcal{L}^+ + \mathcal{L}^- = 0]</td>
<td>[\mathcal{L}^+ + \mathcal{L}^- = 1]</td>
</tr>
<tr>
<td>[\mathcal{L}^- + \mathcal{L}^+ &gt; 1]</td>
<td>[\mathcal{L}^+ + \mathcal{L}^- = 0]</td>
<td>[\mathcal{L}^+ + \mathcal{L}^- = 1]</td>
</tr>
</tbody>
</table>

**Table:**

- \[\mathcal{L}^+ + \mathcal{L}^- = 0\]: CO
- \[\mathcal{L}^+ + \mathcal{L}^- = 1\]: CO, \(\land\)BC toDNF
- \[\mathcal{L}^+ + \mathcal{L}^- > 1\]: CO, \(\land\)BC toDNF, IM

**Notes:**

- \(\land\)BC toDNF
- \(\lor\)BC toCNF
- \(\land\)BC toDNF, CE
- \(\lor\)BC toCNF, CE, \(\land\)BC
Proof System Basic Statements

\[
(\bigcap_{X_i \in X} X_i)[A] \cap \bigcap_{L_i \in \mathcal{L}} L_i \subseteq \bigcup_{L_i' \in \mathcal{L}'} L_i' \quad \text{and} \\
[A](\bigcap_{X_i \in X} X_i) \cap \bigcap_{L_i \in \mathcal{L}} L_i \subseteq \bigcup_{L_i' \in \mathcal{L}'} L_i'.
\]

<table>
<thead>
<tr>
<th>(\mathcal{L}^- + \mathcal{L}'^+)</th>
<th>(\text{CO, } \land \text{BC, CL, RN} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(\text{CO, } \land \text{BC, CL, RN} )</td>
</tr>
<tr>
<td>1</td>
<td>(\text{SE, } \land \text{BC, CL, RN} )</td>
</tr>
<tr>
<td>&gt; 1</td>
<td>(\text{SE, } \lor \text{BC, } \land \text{BC, CL, RN} )</td>
</tr>
</tbody>
</table>

\(\land \text{CNF, CE, } \land \text{BC, CL, RN} \)
Proof System Basic Statements

\[ L \subseteq L' \text{ (mixed):} \]

<table>
<thead>
<tr>
<th>( \varphi_R \models \psi_{R'} )</th>
<th>( \neg \psi_{R'} \models \neg \varphi_R )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \varphi_R \models \psi_{R'} )</td>
<td>( \neg \psi_{R'} \models \neg \varphi_R )</td>
</tr>
<tr>
<td>( \neg \varphi_{R'} \models \psi_{R'} )</td>
<td>( \neg \psi_{R'} \models \varphi_R )</td>
</tr>
<tr>
<td>( \varphi_{R'} \models \neg \psi_{R'} )</td>
<td>( \psi_{R'} \models \neg \varphi_R )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R</th>
<th>R'</th>
</tr>
</thead>
<tbody>
<tr>
<td>ME, ns</td>
<td>MO</td>
</tr>
<tr>
<td>toDNF</td>
<td>IM</td>
</tr>
<tr>
<td>CE</td>
<td>toCNF</td>
</tr>
<tr>
<td>ME</td>
<td>MO, ns</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R</th>
<th>R'</th>
</tr>
</thead>
<tbody>
<tr>
<td>ME, ns</td>
<td>MO, CT</td>
</tr>
<tr>
<td>toCNF</td>
<td>IM</td>
</tr>
<tr>
<td>IM</td>
<td>toCNF</td>
</tr>
<tr>
<td>MO, CT</td>
<td>ME, ns</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R</th>
<th>R'</th>
</tr>
</thead>
<tbody>
<tr>
<td>ME, ns</td>
<td>MO</td>
</tr>
<tr>
<td>toDNF</td>
<td>CE</td>
</tr>
<tr>
<td>CE</td>
<td>toDNF</td>
</tr>
<tr>
<td>MO</td>
<td>ME, ns</td>
</tr>
</tbody>
</table>
### M&S

<table>
<thead>
<tr>
<th>$M^3$</th>
<th>$\mu_0^2$</th>
<th>$\mu_1^2$</th>
<th>$\mu_2^2$</th>
<th>$\mu_3^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha_0^3$</td>
<td>2</td>
<td>$\infty$</td>
<td>0</td>
<td>$\infty$</td>
</tr>
<tr>
<td>$\alpha_1^3$</td>
<td>1</td>
<td>3</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$A^3$</th>
<th>$v_3$ = 0</th>
<th>$v_3$ = 1</th>
<th>$v_3$ = 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A^2$</td>
<td>$\alpha_0^3$</td>
<td>$\alpha_1^3$</td>
<td>$\alpha_0^3$</td>
</tr>
<tr>
<td>$A^1$</td>
<td>$v_1$ = 0</td>
<td>$v_1$ = 1</td>
<td>$v_1$ = 2</td>
</tr>
</tbody>
</table>

\[ v_3 = 0 \] 
\[ v_3 = 1 \] 
\[ v_3 = 2 \]

\[ A^2 \]
\[ v_2 = 0 \] 
\[ v_2 = 1 \]

\[ A^1 \]
\[ v_1 = 0 \] 
\[ v_1 = 1 \] 
\[ v_1 = 2 \]
\[ v_1 = 0 \]
\[ v_1 = 1 \]
\[ v_1 = 2 \]
\[ v_2 = 0 \]
\[ v_2 = 1 \]
\[ v_3 = 0 \]
\[ v_3 = 1 \]
\[ v_3 = 2 \]
failed

Failed FD\textsuperscript{c} runtime (in s)

Failed FD\textsuperscript{p} runtime (in s)

h\textsuperscript{max}

h\textsuperscript{M\&S}

Generation
Witness size in relation to dead-ends

![Graph 1: Witness size vs. percentage of dead-ends](image1)

- y-axis: $\frac{\text{witness size } F_{D_p}}{\text{FD}_c}$
- x-axis: percentage of dead-ends

- Markers: $h_{\text{max}}$, $h_{\text{M&S}}$

![Graph 2: Verifier runtime vs. percentage of dead-ends](image2)

- y-axis: $\frac{\text{verifier runtime } F_{D_p}}{\text{FD}_c}$
- x-axis: percentage of dead-ends

- Markers: $h_{\text{max}}$, $h_{\text{M&S}}$
Future Work

- cover more planning techniques
  - planning as satisfiability
  - potential heuristics
  - partial order reduction
  - ...

- extend witness definition
  - inductive certificates: more compositions
  - proof system: more rules, more general basic statements