Pseudo-Boolean Proof Logging for Optimal Planning

Simon Dold Malte Helmert Jakob Nordström
Gabriele Röger Tanja Schindler

University of Basel
University of Copenhagen and Lund University

WHOOPS 2025

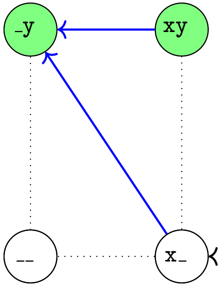
Pseudo-Boolean Proof Logging for Optimal Planning

Planning Task induces Factored Transition System

Variables: $\{x, y \quad \}$
Initial state: $\{x\}$
Goal condition: $\{y \quad \}$
Actions: ⟨pre, add, del, cost⟩

- $a_1 = \langle \{x\}, \{y\}, \{x\}, 1 \rangle$

Plan: Sequence of actions that lead form the initial state to a goal state.

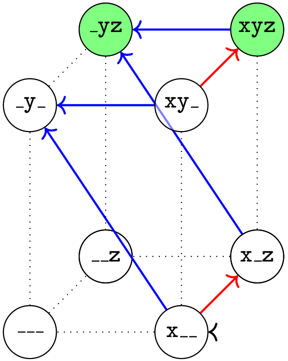


Planning Task induces Factored Transition System

Variables: $\{x, y, z\}$
Initial state: $\{x\}$
Goal condition: $\{y, z\}$
Actions: ⟨pre, add, del, cost⟩

- $a_1 = \langle\{x\}, \{y\}, \{x\}, 1\rangle$
- $a_2 = \langle\{x\}, \{z\}, \{\}, 2\rangle$

Plan: Sequence of actions that lead form the initial state to a goal state.

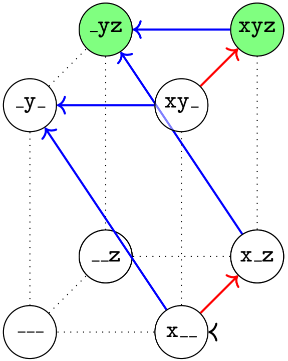


Related to Bounded Model Checking

Variables: $\{x, y, z\}$
Initial state: $\{x\}$
Goal condition: $\{y, z\}$
Actions: $\langle$pre, add, del, cost$\rangle$

- $a_1 = \langle\{x\}, \{y\}, \{x\}, 1\rangle$
- $a_2 = \langle\{x\}, \{z\}, \{\}, 2\rangle$

Plan: Sequence of actions that lead form the initial state to a goal state.

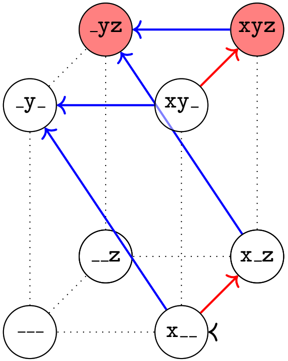


Related to Bounded Model Checking

Variables: $\{x, y, z\}$
Initial state: $\{x\}$
Goal condition: $\{y, z\}$
Actions: ⟨pre, add, del, cost⟩

- $a_1 = \langle\{x\}, \{y\}, \{x\}, 1\rangle$
- $a_2 = \langle\{x\}, \{z\}, \{\}, 2\rangle$

Plan: Sequence of actions that lead form the initial state to a goal state.

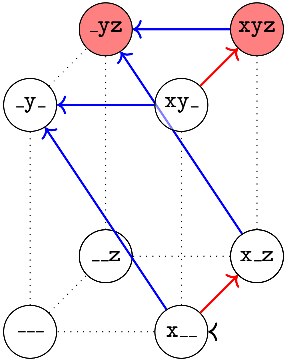


Related to Bounded Model Checking

Variables: $\{x, y, z\}$
Initial states: $\{\{x\}\}$
Goal condition: $\{y, z\}$
Actions: ⟨pre, add, del, cost⟩

- $a_1 = \langle\{x\}, \{y\}, \{x\}, 1\rangle$
- $a_2 = \langle\{x\}, \{z\}, \{\}, 2\rangle$

Plan: Sequence of actions that lead form the initial state to a goal state.

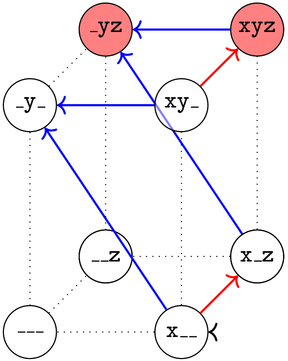


Related to Bounded Model Checking

Variables: $\{x, y, z\}$
Initial states: $\{\{x\}\}$
Error condition: $\{y, z\}$
Actions: ⟨pre, add, del, cost⟩

- $a_1 = \langle\{x\}, \{y\}, \{x\}, 1\rangle$
- $a_2 = \langle\{x\}, \{z\}, \{\}, 2\rangle$

Plan: Sequence of actions that lead form the initial state to a goal state.

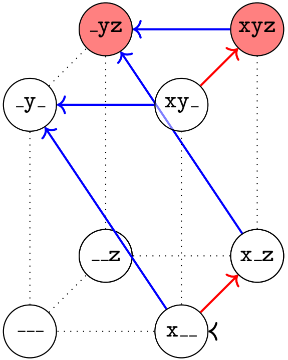


Related to Bounded Model Checking

Variables: $\{x, y, z\}$
Initial states: $\{\{x\}\}$
Error condition: $\{y, z\}$
Actions: ⟨pre, add, del, cost⟩

- $a_1 = \langle\{x\}, \{y\}, \{x\}, 1\rangle$
- $a_2 = \langle\{x\}, \{z\}, \{\}, 2\rangle$

Error trace: Sequence of actions that lead form an initial state to an error state.

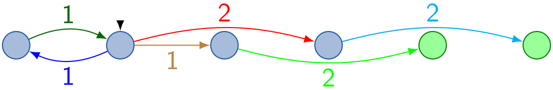


Pseudo-Boolean Proof Logging for Optimal Planning

Optimal Planning

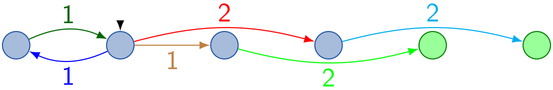
Given: a planning task Π

Output: "$\langle a_1, \ldots, a_n \rangle$ is a plan for Π with minimal cost ",
or "no plan for Π exists".



Pseudo-Boolean Proof Logging for Optimal Planning

Optimal Planning
Given: a planning task Π
Output: "$\langle a_1, \ldots, a_n \rangle$ is a plan for Π with minimal cost", or "no plan for Π exists".



Complexity

- Checking a given plan is in **P**.
- However, plans can be exponentially long.
- Planning is **PSPACE**-complete.
  - A bounded-length computation of a nondeterministic Turing machine can be represented as a planning task.

Pseudo-Boolean Proof Logging for Optimal Planning

Pseudo-Boolean Proof Logging for Optimal Planning

Proofs for planner outputs

- "$\langle a_0, \ldots, a_n \rangle$ is a plan for Π"
  $\rightsquigarrow$ The plan is the proof. Use validator (e.g., VAL, INVAL)
- "no plan for Π exists"
  $\rightsquigarrow$ unsolvability certificate[1]
- "$\langle a_0, \ldots, a_n \rangle$ is a plan for Π with minimal cost"
  $\rightsquigarrow$ lower-bound certificate[2] (and validate plan)

---

[1] Salomé Eriksson. *Certifying Planning Systems: Witnesses for Unsolvability* (Ph.D. Thesis 2019)

[2] Esther Mugdan, Remo Christen and Salomé Eriksson. *Optimality Certificates for Classical Planning* (ICAPS 2023)

Pseudo-Boolean Proof Logging for Optimal Planning

There is no plan with cost lower than B iff there is a property φ over state-cost pairs that

1. holds for the initial state I with cost 0
2. is inductive under action applications (a.k.a. an invariant)
3. and does not hold for a goal state with cost lower than B

Lower-bound certificate: φ + proofs for (1)–(3)

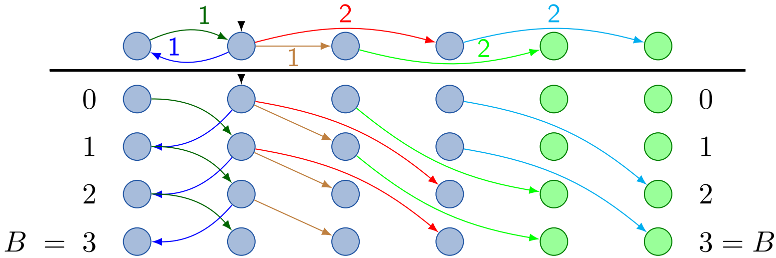
Pseudo-Boolean Proof Logging for Optimal Planning

There is no plan with cost lower than B iff there is a property φ over state-cost pairs that

1. holds for the initial state I with cost 0
2. is inductive under action applications (a.k.a. an invariant)
3. and does not hold for a goal state with cost lower than B

Lower-bound certificate: φ + proofs for (1)–(3)

$\rightsquigarrow$ corresponds to separating set in the state-cost pair graph

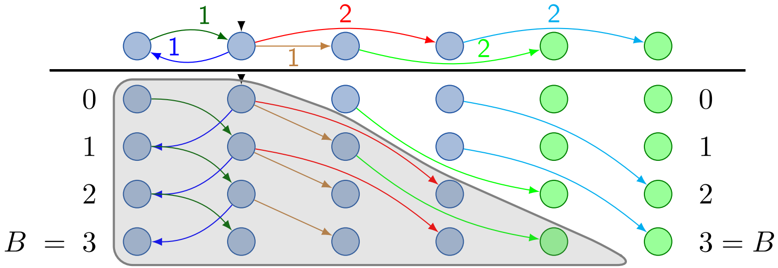


Pseudo-Boolean Proof Logging for Optimal Planning

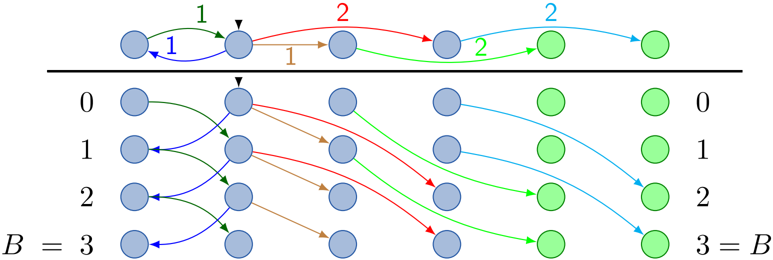
There is no plan with cost lower than B iff there is a property φ over state-cost pairs that

1. holds for the initial state I with cost 0
2. is inductive under action applications (a.k.a. an invariant)
3. and does not hold for a goal state with cost lower than B

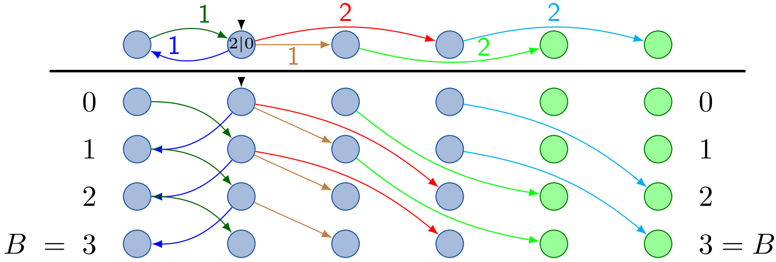
Lower-bound certificate: φ + proofs for (1)–(3)
$\rightsquigarrow$ corresponds to separating set in the state-cost pair graph



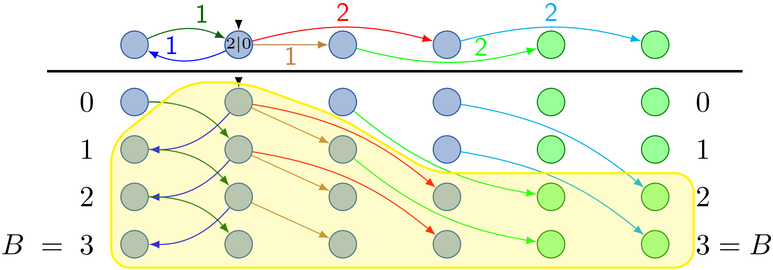
Pseudo-Boolean Proof Logging for Optimal Planning



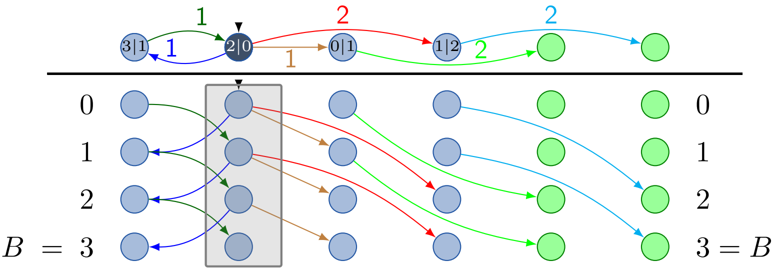
- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- (1|2) indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  $\rightsquigarrow$ Lower-bound certificate for that state



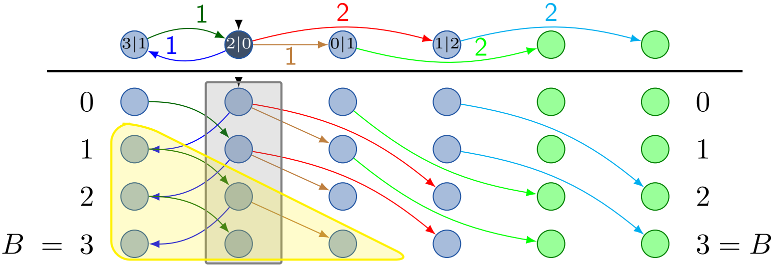
- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- 1|2 indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  $\rightsquigarrow$ Lower-bound certificate for that state



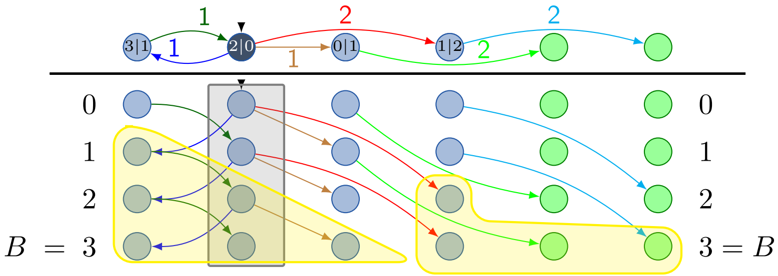
- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- (1|2) indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  $\rightsquigarrow$ Lower-bound certificate for that state



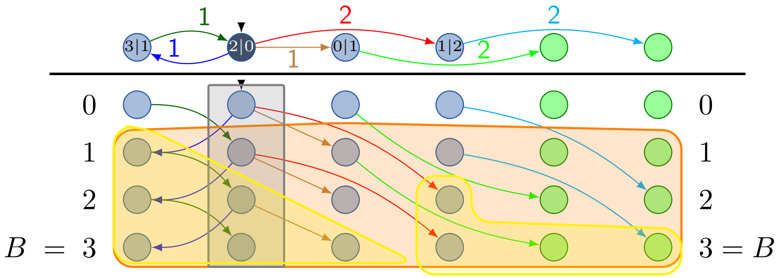
- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- 1|2 indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  $\rightsquigarrow$ Lower-bound certificate for that state



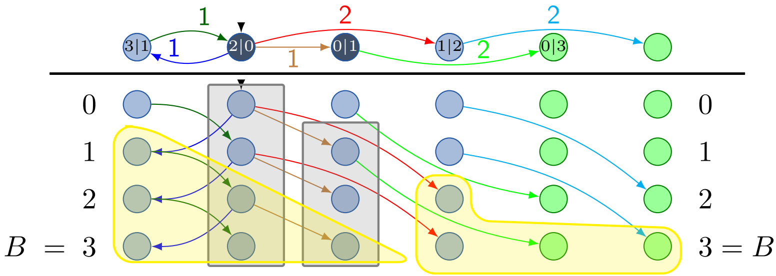
- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- (1|2) indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  $\rightsquigarrow$ Lower-bound certificate for that state



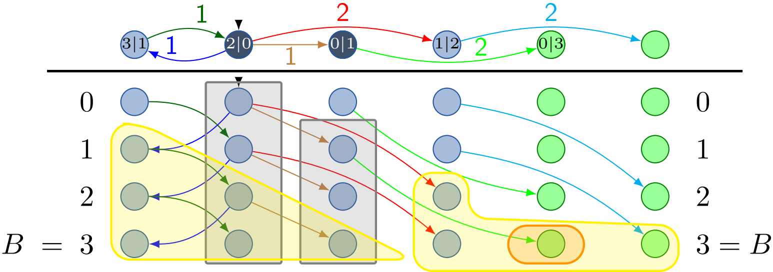
- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- 1|2 indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  $\rightsquigarrow$ Lower-bound certificate for that state



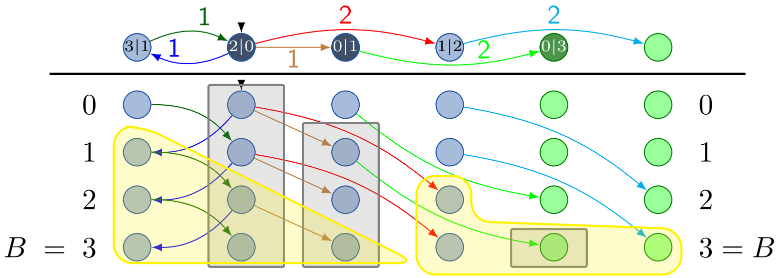
- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- 1|2 indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  $\rightsquigarrow$ Lower-bound certificate for that state



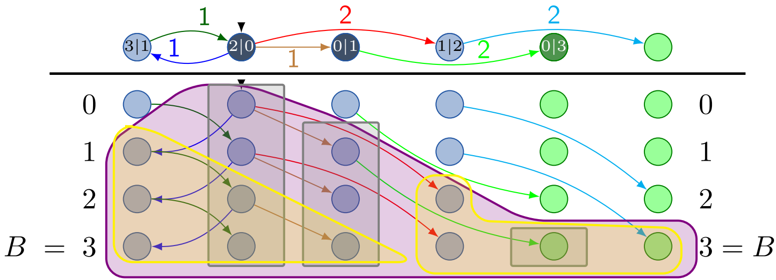
- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- 1|2 indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  $\rightsquigarrow$ Lower-bound certificate for that state



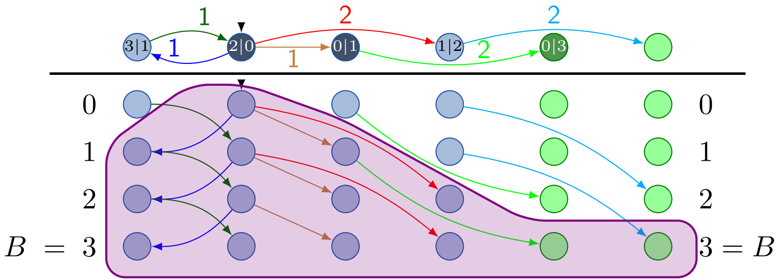
- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- 1|2 indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  $\rightsquigarrow$ Lower-bound certificate for that state



- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- 1|2 indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  $\rightsquigarrow$ Lower-bound certificate for that state



- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- 1|2 indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  $\rightsquigarrow$ Lower-bound certificate for that state



- A*: From all considered paths, take the most promising and also consider its one-step continuations.
- g is the g-value "how much was used to get here?"
- h is the heuristic value "how much more is necessary?"
- Most promising means $h + g$ is minimal.
- 1|2 indicates $h = 1$ and $g = 2$.
- Admissible heurisitc never overestimates
  ⇝ Lower-bound certificate for that state

Pseudo-Boolean Proof Logging for Optimal Planning

Certifying Optimality based on Pseudo-Boolean Constraints

- proof logging:
  - log representation of invariant φ as pseudo-Boolean circuit
  - log pseudo-Boolean constraint proofs for the three properties (initial state, goal, inductivity)
- verification:
  - encode planning semantics as pseudo-Boolean constraints
  - combine with invariant definition and proof log
  - use VeriPB to verify resulting pseudo-Boolean proof

Pseudo-Boolean Encoding of Planning Semantics - Part I

Given: planning task $\Pi = \langle V, I, G, A \rangle$

Encoding: (similar to SAT encoding with horizon 1)

- Boolean state variables V:
  PB variables V, PB cost variables $V_c = \{c_0, \dots, \lceil \log_2 B \rceil\}$,
  copies V', V'_c
- initial state $I \subseteq V$:

$$r_I \Leftrightarrow \sum_{v \in I} v + \sum_{v \in V \setminus I} \bar{v} \geq |V|$$

- goal $G \subseteq V$:

$$r_G \Leftrightarrow \sum_{v \in G} v \geq |G|$$

Pseudo-Boolean Encoding of Planning Semantics - Part II

- actions $a \in A$ with preconditions $\mathit{pre}(a) \subseteq V$, add effects $\mathit{add}(a) \subseteq V$, delete effects $\mathit{del}(a) \subseteq V$, cost $\mathit{cost}(a) \in \mathbb{N}_0$:

$$r_a \Rightarrow \sum_{v \in \mathit{pre}(a)} v + \sum_{v \in \mathit{add}(a)} v' + \sum_{v \in \mathit{del}(a)} \overline{v'} + \sum_{v \in V \setminus \mathit{evars}(a)} eq_{v,v'} + \Delta c^{=\mathit{cost}(a)} \geq |\mathit{pre}(a)| + |V| + 1$$

where (here the Pseudo-Boolean encoding is very useful)

$$\Delta c^{=k} \Leftrightarrow \sum_{i=0}^{\lceil \log_2 B \rceil} 2^i c'_i - \sum_{i=0}^{\lceil \log_2 B \rceil} 2^i c_i = k$$

- transition relation:

$$r_T \Leftrightarrow \sum_{a \in A} r_a \geq 1$$

Pseudo-Boolean Lower Bound Certificates

Lower-bound certificate for Π with bound B:

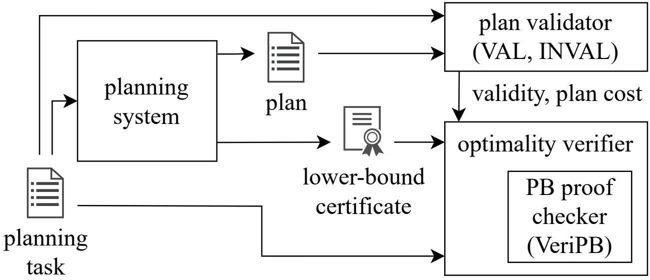
- PB circuit representing invariant φ based on variables V, V_c:

$$r_0 :\Leftrightarrow C(V, V_c)$$
$$\dots$$
$$r_n :\Leftrightarrow C(V, V_c, r_0, \dots, r_{n-1})$$
$$r_\varphi :\Leftrightarrow C(V, V_c, r_0, \dots, r_{n-1}, r_n)$$

- VeriPB proof for initial state lemma $\overline{r_I} + \overline{\mathit{cost}_{=0}} + r_\varphi \geq 1$
- VeriPB proof for goal lemma $\overline{r_G} + \overline{r_\varphi} + \mathit{cost}_{\geq B} \geq 1$
- VeriPB proof for inductivity lemma $\overline{r_\varphi} + \overline{r_T} + r'_\varphi \geq 1$

Note: VeriPB proof contains two synchronized copies (unprimed+primed) of the circuit reifications (and some proof parts)

Certified Optimal Planning



Current Status

Current Status

General framework:

- definition of pseudo-Boolean lower-bound certificates ✔
- PB encoding of planning semantics ✔
  formally verified ✘
- implementation ✘ (WIP)
- theoretical relation to earlier approaches ✘ (WIP)

Proof logging planning algorithms:

- general approach for heuristic search ✔
- PDB and h^{max} heuristics ✔
- implementation ✘ (WIP)
- more heuristics ✘
- SAT planning, symbolic search ✘

⇝ more details in our arXiv/ICAPS 2025 paper