Verifying Unsolvability in Classical Planning with VeriPB

Simon Dold Tanja Schindler
Jakob Nordström Malte Helmert

University of Basel, University of Copenhagen

AVM 2024

Planning

Planning Task $\Pi = \langle V, I, A, \gamma \rangle$

- state variables V
- initial state I
- actions A
- goal γ

Find a sequence of actions that leads from the initial state to a state that matches the goal

Planning

Planning Task $\Pi = \langle V, I, A, \gamma \rangle$

- state variables $V = \{money, cake, eaten\}$
- initial state I
- actions A
- goal γ

Find a sequence of actions that leads from the initial state to a state that matches the goal

Planning

Planning Task $\Pi = \langle V, I, A, \gamma \rangle$

- state variables $V = \{money, cake, eaten\}$
- initial state $I = \{money, \overline{cake}, \overline{eaten}\}$
- actions A
- goal γ

Find a sequence of actions that leads from the initial state to a state that matches the goal

Planning

Planning Task $\Pi = \langle V, I, A, \gamma \rangle$

- state variables $V = \{money, cake, eaten\}$
- initial state $I = \{money, \overline{cake}, \overline{eaten}\}$
- actions $A = \{buy, sell, eat\}$
- goal γ

Find a sequence of actions that leads from the initial state to a state that matches the goal

Planning

Planning Task $\Pi = \langle V, I, A, \gamma \rangle$

- state variables $V = \{money, cake, eaten\}$
- initial state $I = \{money, \overline{cake}, \overline{eaten}\}$
- actions $A = \{buy, sell, eat\}$
- goal $\gamma = \{cake, eaten\}$

Find a sequence of actions that leads from the initial state to a state that matches the goal

Planning

Each action has a precondition, add effect, delete effect

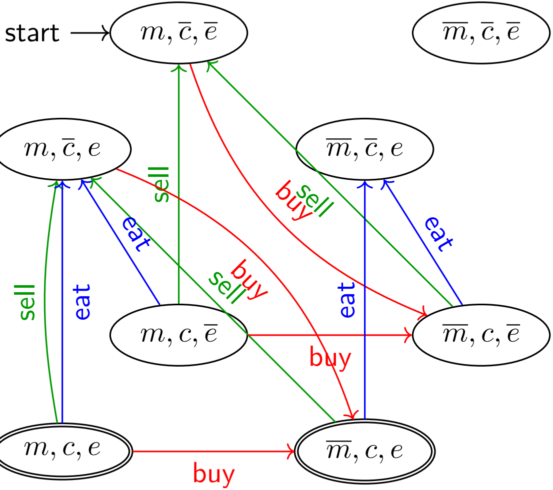
	pre	add	del
buy	{money}	{cake}	{money}
sell	{cake}	{money}	{cake}
eat	{cake}	{eaten}	{cake}

Planning

Task induces a directed graph called state space

Planning

Task induces a directed graph called state space



Verification of the Planner Result

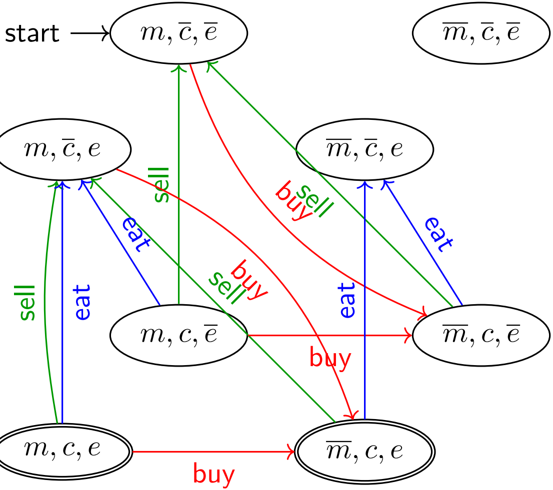
Use a planner to find a sequence that leads from the initial state to the goal. MyPlanner 1.0:

- $[buy, eat, buy]$

Don't trust it? Verify it by executing the plan.

Verification of the Planner Result

Plan: $[buy, eat, buy]$



Verification of the Planner Result

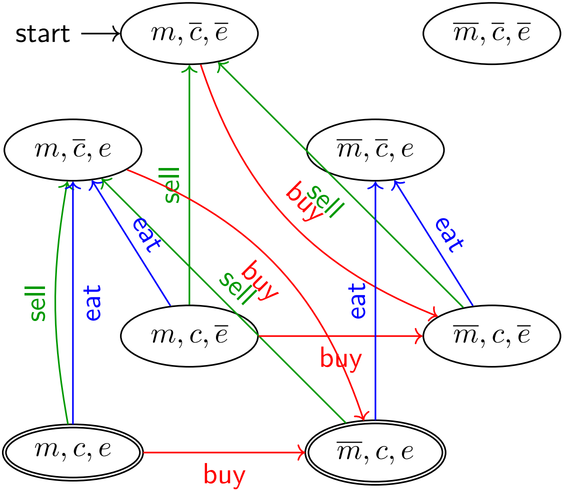
Use another planner to find a sequence that leads from the initial state to the goal. MyPlanner 2.0:

- unsolvable

How to verify? With a Certificate.

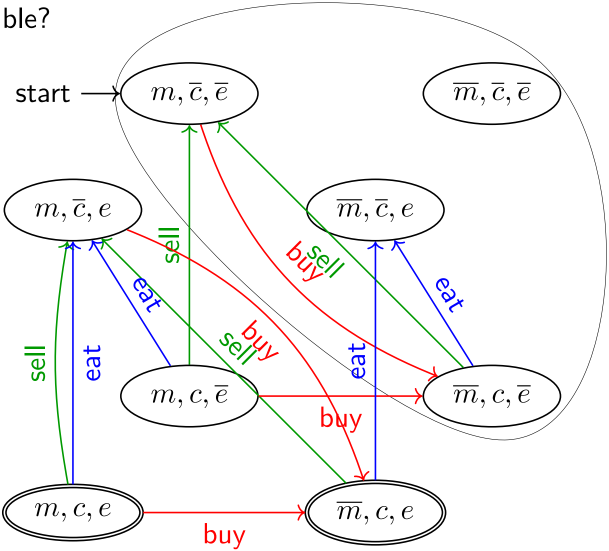
Verification of the Planner Result

unsolvable?



Verification of the Planner Result

unsolvable?



Inductive Set

Inductive Set

If state s is a member of inductive set φ, then all successors of s are members of φ, too.

Idea based on Salomé Eriksson
Certifying Planning Systems: Witnesses for Unsolvability (2019)

Idea

- Find a set φ (with compact representation) and prove that
  - φ contains the initial state
  - φ is an inductive set
  - φ contains no goal state
- Such a set φ exists if and only if the task is unsolvable.

Idea

- Translate the planning task and φ into pseudo boolean constraints (PBCs) $\sum a_i \cdot \ell_i \geq A$ with literals ℓ_i and $a_i, A \in \mathbb{N}$.
- $2 \cdot \varphi + money + cake + eaten \geq 2$
- $\overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$

Idea

- Use extended cutting plane proof system to deduce further constraints by
  - Addition of two PBCs.
  - Multiplication of a PBC.
  - Division of a PBC with rounding up.
  - Reification (introducing auxillary variables).

Idea

- We construct φ and the proof during the search.
  - For each search technique we have to come up with a fitting algorithm.
- This proof does not certify the planner.
- Only certifies this particular result.
- It can be checked by an independent party that knows nothing about planning.
  - For this we use VeriPB.[1]

[1] Bart Bogaerts, Stephan Gocht, Ciaran McCreesh, Jakob Nordström *Certified Dominance and Symmetry Breaking for Combinatorial Optimisation.* (JAIR 2023)

VeriPB

- Checks proof in the extended cutting plane proof system.
- The core checker (CakePB[2]) is verified by HOL4 [3] .

[2] Bart Bogaerts, Ciaran McCreesh, Magnus O. Myreen, Jakob Nordström, Andy Oertel, and Yong Kiam Tan.
Documentation of VeriPB and CakePB for the SAT Competition 2023
(SAT Competition 2023)

[3] Norrish, M., Slind, K.: *HOL-4 manuals* (1998-2008), `http://hol.sourceforge.net/`

Outlook

- Next steps
  - Implement task to PBCs translation
  - Implement proof logging for naive search
- Future steps
  - Certify optimality if actions have cost
  - Argue with fancier algorithms
    - Abstractions, relaxations
    - Admissible heuristics

Example subproof

- (1) from framework: $\overline{money} \geq 0$
- (2) from task: $3 \cdot \overline{I} + money + \overline{cake} + \overline{eaten} \geq 3$
- (3) from φ: $2 \cdot \varphi + money + cake + eaten \geq 2$

Example subproof

- (1) from framework: $\overline{money} \geq 0$
- (2) from task: $3 \cdot \overline{I} + money + \overline{cake} + \overline{eaten} \geq 3$
- (3) from φ: $2 \cdot \varphi + money + cake + eaten \geq 2$
- (4) via (2)+(3): $3 \cdot \overline{I} + 2 \cdot \varphi + 2 \cdot money + 1 + 1 \geq 5$
- (5) via (4)+2·(1): $3 \cdot \overline{I} + 2 \cdot \varphi \geq 1$
- (6) via (5)/3: $\lceil \frac{3}{3} \rceil \cdot \overline{I} + \lceil \frac{2}{3} \rceil \cdot \varphi \geq \lceil \frac{1}{3} \rceil$

$\overline{I} + \varphi \geq 1$

Example subproof

In VeriPB this would look like:

- `1 ~money >= 0;`
- `3 ~I 1 money 1 ~cake 1 ~eaten >= 3;`
- `2 phi 1 money 1 cake 1 eaten >= 2;`

Example subproof

In VeriPB this would look like:

- `1 ~money >= 0;`
- `3 ~I 1 money 1 ~cake 1 ~eaten >= 3;`
- `2 phi 1 money 1 cake 1 eaten >= 2;`
- `pol 2 3 + 1 2 * + 3 d`
- `e 1 ~I 1 phi >= 1 ; -1`

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

- $\varphi \geq 1$, $buy \geq 1$, $\overline{\varphi'} \geq 1$ via assumption

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

- $\varphi \geq 1$, $buy \geq 1$, $\overline{\varphi'} \geq 1$ via assumption

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

- $\varphi \geq 1$, $buy \geq 1$, $\overline{\varphi'} \geq 1$ via assumption
- $money \geq 1$, $\overline{money'} \geq 1$, $eq_{eaten} \geq 1$ via (1)

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

- $\varphi \geq 1$, $buy \geq 1$, $\overline{\varphi'} \geq 1$ via assumption
- $money \geq 1$, $\overline{money'} \geq 1$, $eq_{eaten} \geq 1$ via (1)

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

- $\varphi \geq 1$, $buy \geq 1$, $\overline{\varphi'} \geq 1$ via assumption
- $money \geq 1$, $\overline{money'} \geq 1$, $eq_{eaten} \geq 1$ via (1)
- $\overline{cake} \geq 1$, $\overline{eaten} \geq 1$ via (2)

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

- $\varphi \geq 1$, $buy \geq 1$, $\overline{\varphi'} \geq 1$ via assumption
- $money \geq 1$, $\overline{money'} \geq 1$, $eq_{eaten} \geq 1$ via (1)
- $\overline{cake} \geq 1$, $\overline{eaten} \geq 1$ via (2)

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

- $\varphi \geq 1$, $buy \geq 1$, $\overline{\varphi'} \geq 1$ via assumption
- $money \geq 1$, $\overline{money'} \geq 1$, $eq_{eaten} \geq 1$ via (1)
- $\overline{cake} \geq 1$, $\overline{eaten} \geq 1$ via (2)
- $\overline{eaten'} \geq 1$ via (3)

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

- $\varphi \geq 1$, $buy \geq 1$, $\overline{\varphi'} \geq 1$ via assumption
- $money \geq 1$, $\overline{money'} \geq 1$, $eq_{eaten} \geq 1$ via (1)
- $\overline{cake} \geq 1$, $\overline{eaten} \geq 1$ via (2)
- $\overline{eaten'} \geq 1$ via (3)

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

- $\varphi \geq 1$, $buy \geq 1$, $\overline{\varphi'} \geq 1$ via assumption
- $money \geq 1$, $\overline{money'} \geq 1$, $eq_{eaten} \geq 1$ via (1)
- $\overline{cake} \geq 1$, $\overline{eaten} \geq 1$ via (2)
- $\overline{eaten'} \geq 1$ via (3)
- $\varphi' \geq 1$ via (4)

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

- $\varphi \geq 1$, $buy \geq 1$, $\overline{\varphi'} \geq 1$ via assumption
- $money \geq 1$, $\overline{money'} \geq 1$, $eq_{eaten} \geq 1$ via (1)
- $\overline{cake} \geq 1$, $\overline{eaten} \geq 1$ via (2)
- $\overline{eaten'} \geq 1$ via (3)
- $\varphi' \geq 1$ via (4)

Example subproof with RUP

- (1): $4 \cdot \overline{buy} + money + cake' + \overline{money'} + eq_{eaten} \geq 4$
- (2): $2 \cdot \overline{\varphi} + \overline{money} + \overline{cake} + \overline{eaten} \geq 2$
- (3): $\overline{eq_{eaten}} + eaten + \overline{eaten'} \geq 1$
- (4): $2 \cdot \varphi' + money' + cake' + eaten' \geq 2$

RUP with assumption $\varphi + buy + \overline{\varphi'} \geq 3$

- $\varphi \geq 1$, $buy \geq 1$, $\overline{\varphi'} \geq 1$ via assumption
- $money \geq 1$, $\overline{money'} \geq 1$, $eq_{eaten} \geq 1$ via (1)
- $\overline{cake} \geq 1$, $\overline{eaten} \geq 1$ via (2)
- $\overline{eaten'} \geq 1$ via (3)
- $\varphi' \geq 1$ via (4)

$\overline{\varphi} + \overline{buy} + \varphi' \geq 1$

Example subproof with RUP

In VeriPB this would look like

- `rup 1 ~phi 1 ~buy 1 phi' >= 1 ;`