Formal Representations of Classical Planning Domains

Claudia Grundke Gabriele Röger Malte Helmert

University of Basel

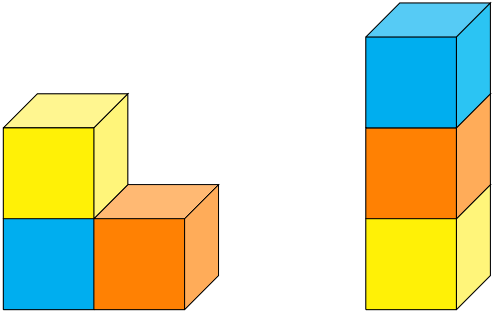
June 4, 2024

Blocksworld

on(x, y)

ontable(x)

clear(x)

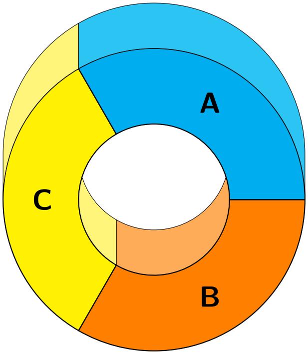


Blocksworld?

$on(A, B)$

$on(B, C)$

$on(C, A)$



Who cares?

Domain Description + ~~Task Description~~

- Generalized Planning (e. g. Srivastava et al., 2011)
- Automated Instance Generation
- Reasoning over Domains

Aims

We want

- a formalism to describe domains precisely,
- it to be based on PDDL,
- to efficiently check if a task belongs to a domain.

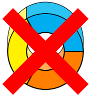
PDDL Axioms

$$\mathit{above}(x,y) \leftarrow \mathit{on}(x,y) \lor \exists z\,(\mathit{on}(x,z) \land \mathit{above}(z,y))$$

PDDL Axioms

$$\mathit{above}(x, y) \leftarrow \mathit{on}(x, y) \lor \exists z\,(\mathit{on}(x, z) \land \mathit{above}(z, y))$$

$$\mathit{illegal}() \leftarrow \exists x\, \mathit{above}(x, x)$$
$$\mathit{illegal}() \leftarrow \dots$$

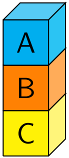


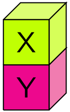
Fixed Goal Condition

All tasks of a domain share same first-order goal.

Example (full-ADL Miconic): $\forall p\,(\textit{passenger}(p) \rightarrow \textit{served}(p))$

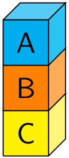
Moving Goal into Initial State

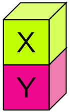




Goal:	$\mathit{on}(A, B) \land \mathit{on}(B, C)$	$\mathit{on}(X, Y)$
New static atoms:	$\mathit{on}^{\mathrm{g}}(A, B), \mathit{on}^{\mathrm{g}}(B, C)$	$\mathit{on}^{\mathrm{g}}(X, Y)$

Moving Goal into Initial State





Goal:	~~$\mathit{on}(A, B) \wedge \mathit{on}(B, C)$~~	~~$\mathit{on}(X, Y)$~~
New static atoms:	$\mathit{on}^{\mathrm{g}}(A, B), \mathit{on}^{\mathrm{g}}(B, C)$	$\mathit{on}^{\mathrm{g}}(X, Y)$

Shared goal:

$$\forall x, x' (\mathit{on}^{\mathrm{g}}(x, x') \rightarrow \mathit{on}(x, x'))$$

Aims

We want

- a formalism to describe domains precisely,
- it to be based on PDDL,
- to efficiently check if a task belongs to a domain.

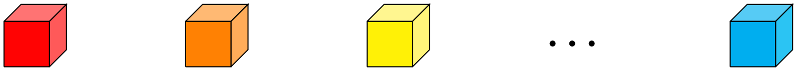
Aims

We want

- a formalism to describe domains precisely,
- it to be based on PDDL,
- to efficiently check if a task belongs to a domain.
- **Maximal expressiveness.**

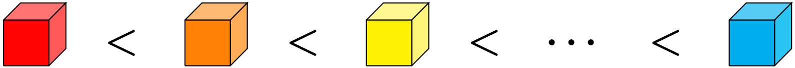
Linear Order

Even number of blocks?



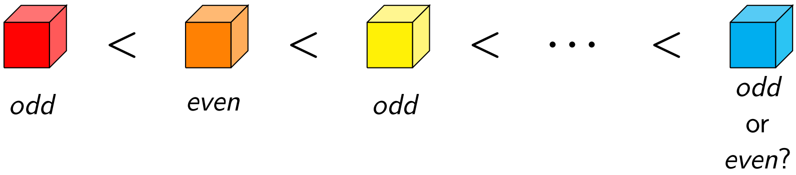
Linear Order

Even number of blocks?



Linear Order

Even number of blocks?



illegal if largest block is *odd*

Capturing P

With linear orders our formalism (using PDDL axioms) can express any polynomial-time algorithm that can decide if a task is legal. (Immerman-Vardi Theorem)

Summary

Link to our paper
Formal Representations of Classical Planning Domains

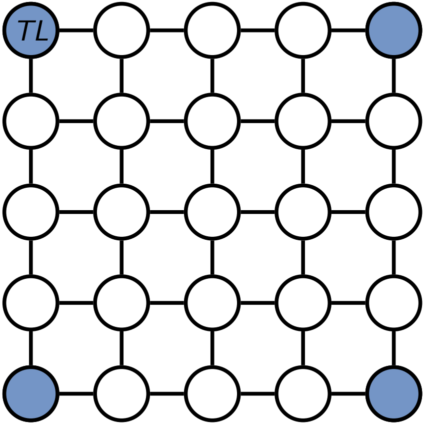
Using PDDL axioms we can restrict which states are legal initial states for tasks of a domain.

- Given a linear order our formalism captures polynomial-time decision algorithms.

All tasks of a domain share same first-order goal.

- Move conjunctive goals into initial state to preserve them.

Linear Order: Grid Graph



$$corner(x) \leftarrow degree_{\geq 2}(x) \land \neg degree_{\geq 3}(x)$$

$$TL(x) \leftarrow corner(x) \land \neg\exists y \, (corner(y) \land y < x)$$