Constrained Pattern Databases

Takudzwa Togarepi, University of Basel, April 14, 2025

Outline

Motivation

- Task - stack blocks in a given order.

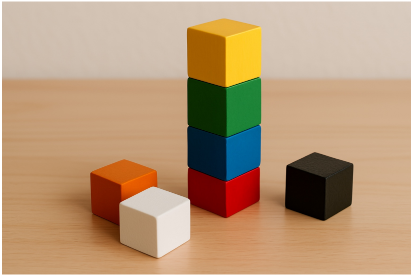


Image generated by *DALL-E*

Motivation

- Task - stack blocks in a given order.
- More blocks, more complex.

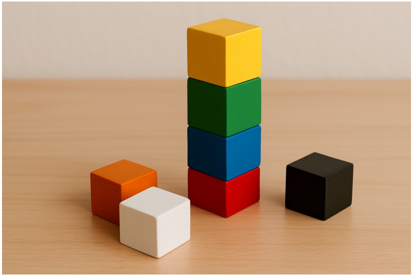


Image generated by *DALL-E*

Motivation

- Task - stack blocks in a given order.
- More blocks, more complex.
- Solution - solve subtasks and combine solutions.

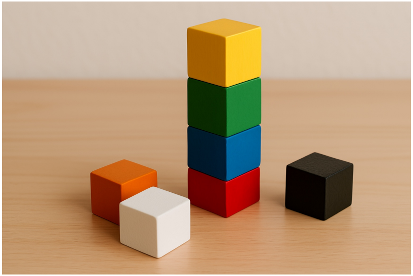


Image generated by *DALL-E*

Outline

Planning tasks

SAS$^+$ Planning Tasks $\Pi = \langle V, O, I, G \rangle$

- Set of variables $V = \{\mathrm{v}_1, \mathrm{v}_2, \mathrm{v}_3, \ldots, \mathrm{v}_n\}$.
- A state s is a full assignment to variables V.
- Finite set of operators O where operator $o = \langle pre(o), eff(o) \rangle$.
- I is the initial state.
- Goal condition G is a partial assignment.

Additional definitions

- A *fact* is a variable-value pair (v_i, d_i).
- A *plan* is a sequence of operators $\pi = \langle o_1, o_2, o_1, o_3 \rangle$.
- An *optimal plan* is one with the minimum cost.

Planning tasks

Abstraction

Given SAS$^+$ task $\Pi = \langle V, O, I, G \rangle$ with state set S and abstraction mapping $\alpha : S \to S'$.

Heuristic h(s)

- $h^\alpha(s)$ estimate of how far state s from the goal.

Abstraction heuristics h^α(s)

- $h^\alpha(s)$ is the distance from s to the nearest goal in the abstract search space.

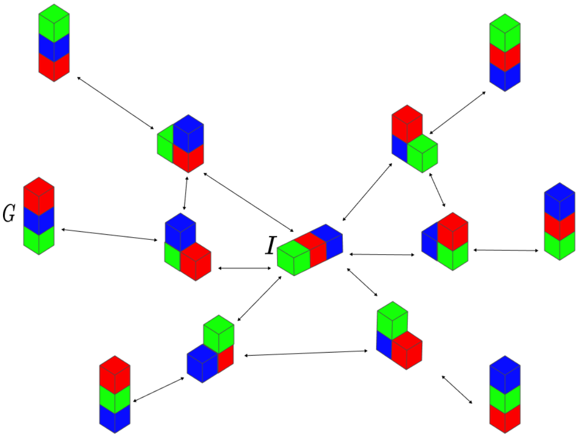
Planning tasks

Projection

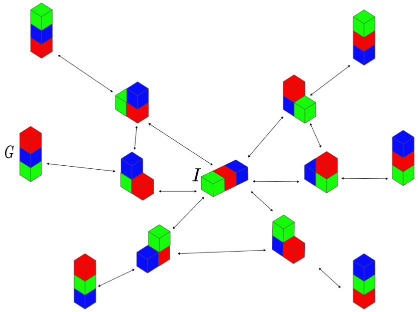
A type of abstraction.
Given SAS$^+$ task $\Pi = \langle V, O, I, G \rangle$ with state set S.
Let $\mathcal{P}$ be a pattern where $\mathcal{P} \subseteq V$.

- $\alpha^{\mathcal{P}}(s) = \alpha^{\mathcal{P}}(s')$ iff $s(p) = s'(p) \mid \forall p \in \mathcal{P}$, where $s, s' \in S$

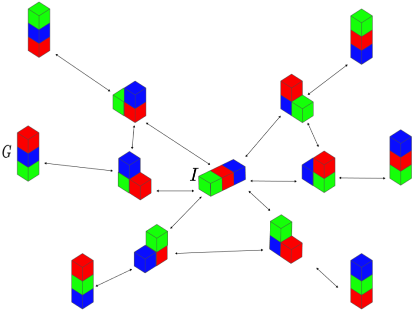
Blocksworld example

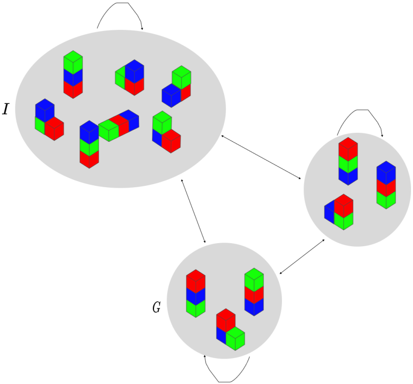


Example of a pattern projection



Example of a pattern projection





pattern = red block

Outline

Pattern Databases

Pattern Database $\text{PDB}(\Pi^{\mathcal{P}}) = \{s_{\mathcal{P}}, d(s_{\mathcal{P}}, G_{\mathcal{P}})\}$

- $s_{\mathcal{P}}$ abstract state .
- $G_{\mathcal{P}}$ abstract goal condition.
- $d(s_{\mathcal{P}},_{\mathcal{P}})$ cost of the cheapest path from $s_{\mathcal{P}}$ to any goal state in the abstract state space.

Outline

Mutexes

Mutex

Two facts (v_i, d_i) and (v_j, d_j) are mutually exclusive (mutex) iff no reachable state s exists such that: $s(v_i) = d_i$ and $s(v_j) = d_j$.

Mutex group

A set of facts in which at most one of them can be true at any point.

Mutexes

Sources of mutexes

1. Fast downward mutexes.
   - get for free from Fast Downward.
2. Fact-alternating mutex groups (Fam-group).
   - uses LPs to infer mutexes.
3. h^2 preprocessor.
   - get for free from translation using h^2.

Outline

Constrained pattern databases (CPDBs)

State constraint

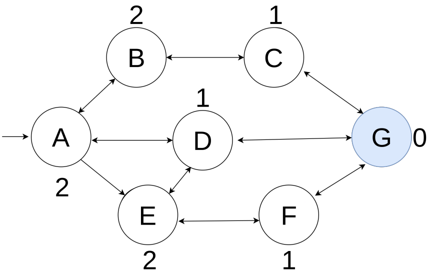
Let $\mathcal{C}$ be a set of mutex groups $\{\mathcal{C}_1, \mathcal{C}_2 \ldots \mathcal{C}_n\}$then :

- $|\mathcal{C}_i \cap s| \leq 1$ holds for all $\mathcal{C}_i \in \mathcal{C}$.

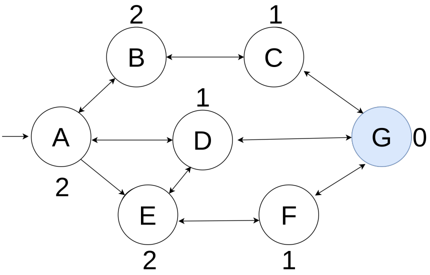
Transition constraint

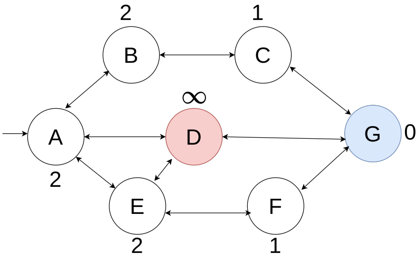
$|(s' \cup \mathrm{pre}(o)) \cap \mathcal{C}_i| \leq 1$ holds for every s' and operator o applicable to s'.

Example of constrained abstract transition system



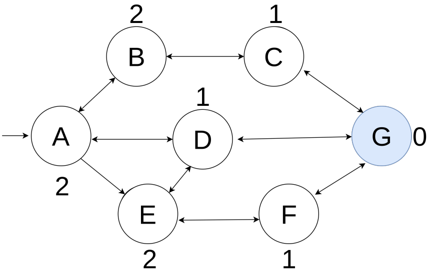
Example of constrained abstract transition system

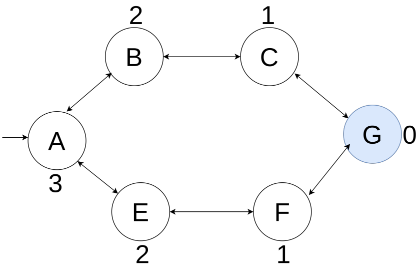




Invalid abstract state

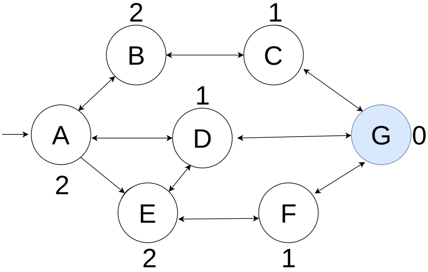
Example of constrained abstract transition system

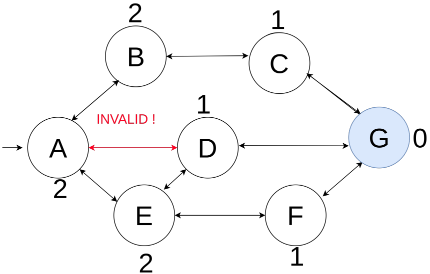




h(A) is 3 now

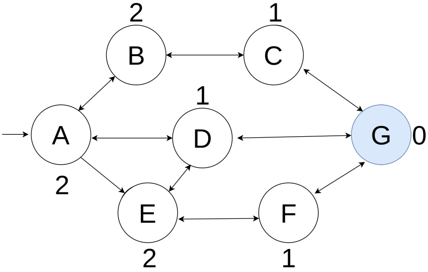
Example of constrained abstract transition system

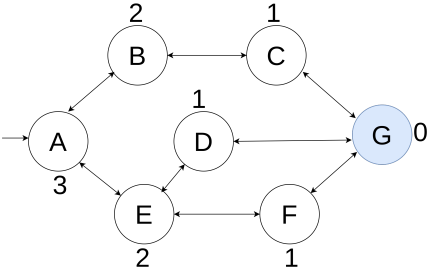




Invalid transition

Example of constrained abstract transition system





h(A) is now 3

Outline

Experiment setup

Benchmarking

- Implemented on Fast Downward .
- Experiments were tested on ICP benchmarks (1847 tasks).
- Time limit 30 minutes per task.
- Memory limit 3584MB.

Experiment setup

Measured attributes

- Coverage.
- Number of failed runs (and reason for failing).
- Pattern database computation time.
- Search time.
- Planner wall clock time.

Experiment setup

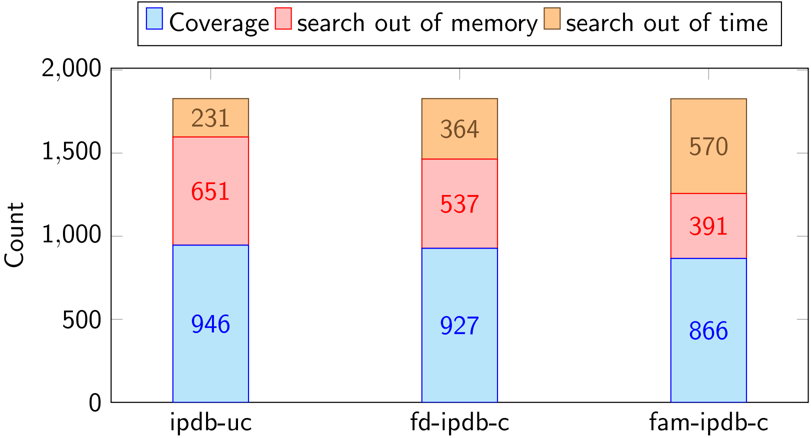
Configurations for ipdb

- baseline ipdb-uc.
- fd-ipdb-c.
- fam-ipdb-c.
- baseline h2-ipdb-uc.
- h2-ipdb-c.

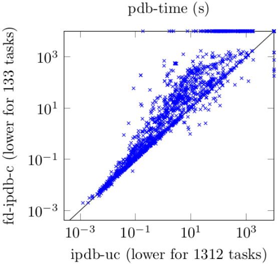
2 different baselines

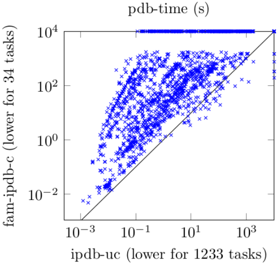
Outline

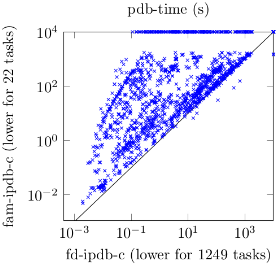
Summary of configurations without h2 preprocessor



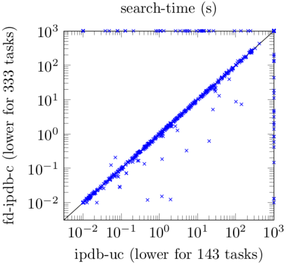
Pattern database computation time

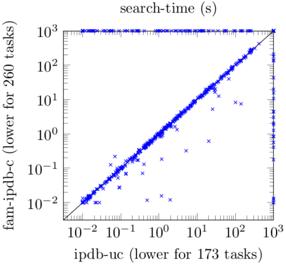


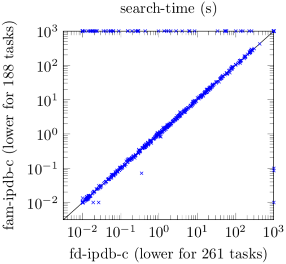




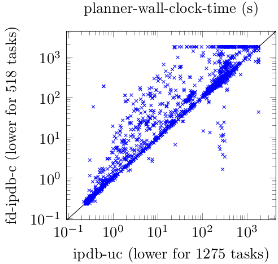
Search time

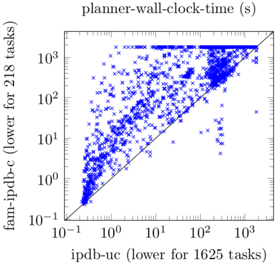

search-time (s)
fd-ipdb-c (lower for 333 tasks)
ipdb-uc (lower for 143 tasks)
10^{-2} 10^{-1} 10^{0} 10^{1} 10^{2} 10^{3}

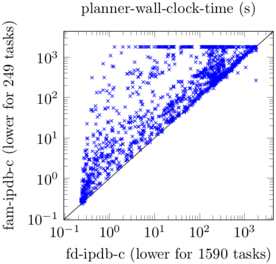

search-time (s)
fam-ipdb-c (lower for 260 tasks)
ipdb-uc (lower for 173 tasks)
10^{-2} 10^{-1} 10^{0} 10^{1} 10^{2} 10^{3}


search-time (s)
fam-ipdb-c (lower for 188 tasks)
fd-ipdb-c (lower for 261 tasks)
10^{-2} 10^{-1} 10^{0} 10^{1} 10^{2} 10^{3}

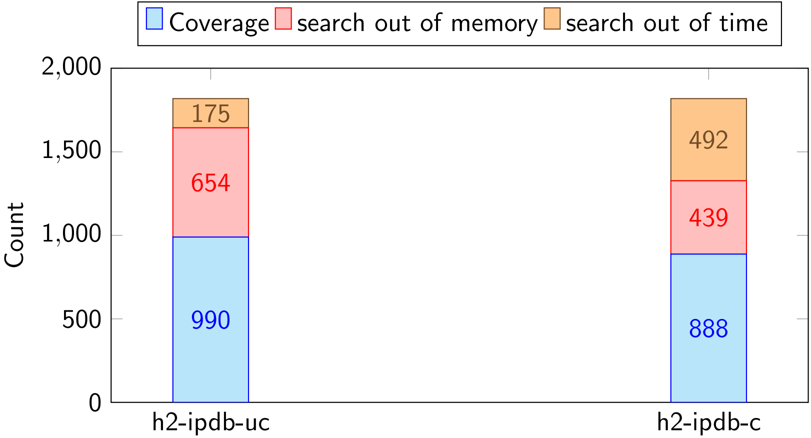
Planner wall clock time



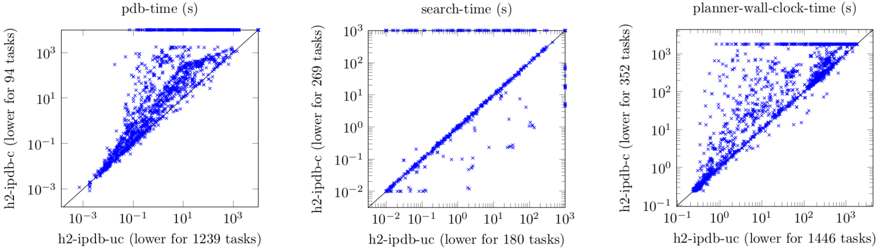




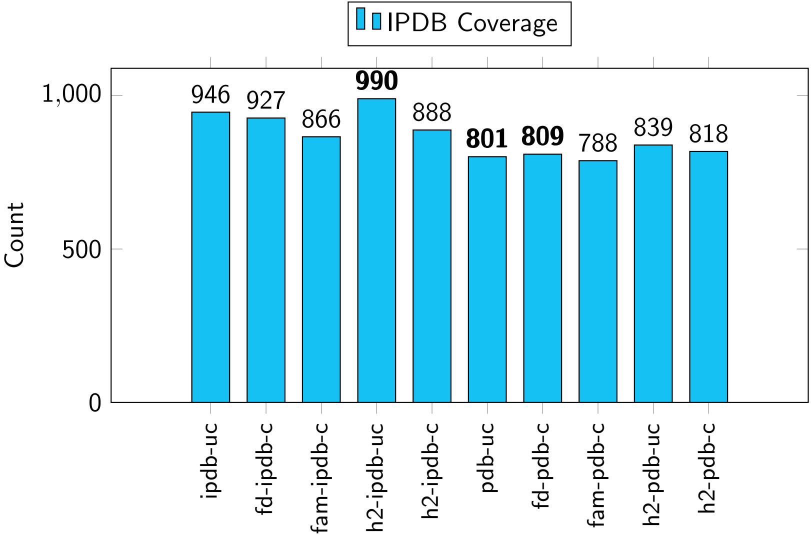
Summary of configurations with h2 preprocessor



h2


pdb-time (s)
h2-ipdb-c (lower for 94 tasks)
h2-ipdb-uc (lower for 1239 tasks)
search-time (s)
h2-ipdb-c (lower for 269 tasks)
h2-ipdb-uc (lower for 180 tasks)
planner-wall-clock-time (s)
h2-ipdb-c (lower for 352 tasks)
h2-ipdb-uc (lower for 1446 tasks)

Coverage for all configurations



Outline

Conclusion

- Fast Downward mutexes offers best trade-off.
- Fam-groups are almost never benefitial.
- Performance of CPDBs and PDBs is close.
- We can potentially improve CPDB coverage by an optimized implementation of CPDBs.

University of Basel

Thank you

for your attention.