An Automata-Based Constraint Programming Framework for Optimal Classical Planning

Damien Van Meerbeeck, Arnaud Lequen, Gilles Pesant, Jendrik Seipp

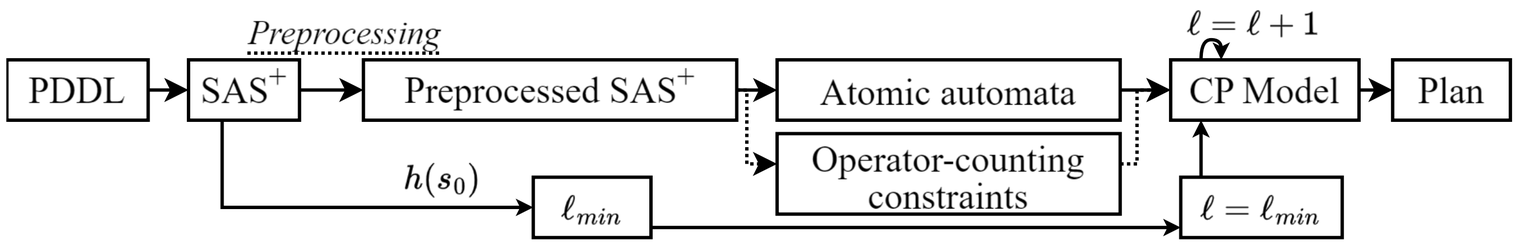
Classical Planning

Optimal Classical Planning is the challenge of finding a shortest sequence of actions (**plan**) transforming an initial world into one that satisfies a goal condition.

Motivation

- Compact representation to better scale for large planning tasks.
- Non-sequential generation of the plan.
- Use of declarative knowledge for redundant constraints.

End-to-End Framework



Input planning task

Planning tasks are generally defined using the standard **PDDL** format. The pipeline first grounds this lifted representation in a **SAS⁺** with the required **state variables**, **actions**, **initial state** and **goal** conditions.

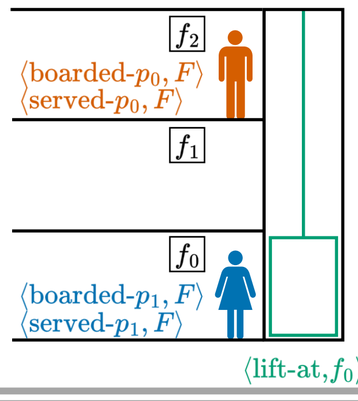
State Variables

lift-at: $\{f_0, f_1, f_2\}$
boarded-p_0: $\{T, F\}$
served-p_0: $\{T, F\}$
boarded-p_1: $\{T, F\}$
served-p_1: $\{T, F\}$

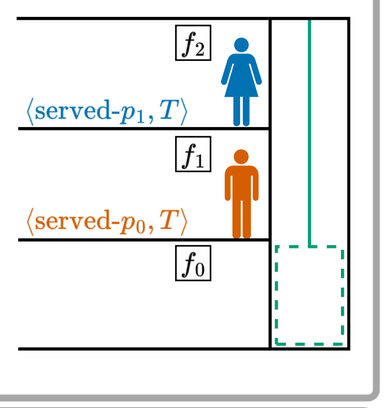
Actions (Σ)

$\text{up}(f_0, f_1)$ $\text{board}(f_2, p_0)$
$\text{up}(f_0, f_2)$ $\text{depart}(f_1, p_0)$
$\text{up}(f_1, f_2)$ $\text{board}(f_0, p_1)$
$\text{down}(f_1, f_0)$ $\text{depart}(f_2, p_1)$
$\text{down}(f_2, f_0)$
$\text{down}(f_2, f_1)$

Initial state s_0

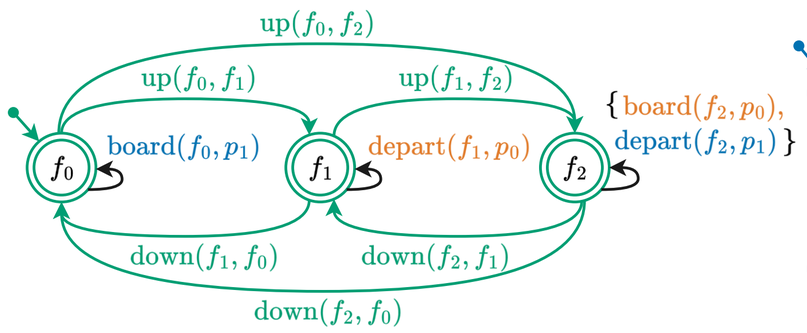


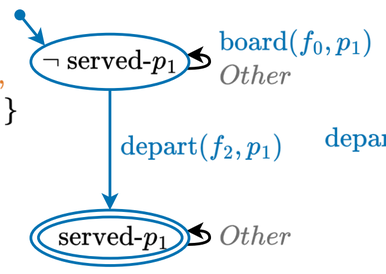
Goal state

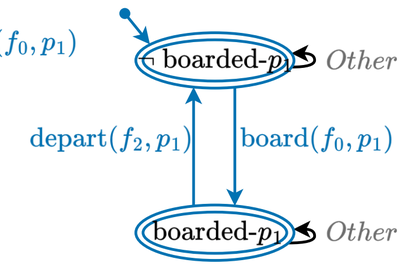


Atomic automata

An automaton is initialized for each state variable by projection of the actions. This results in a **compact factored representation** of the task. Additionally, the alphabet of each automaton is reduced by **grouping equivalent actions** under a single label.

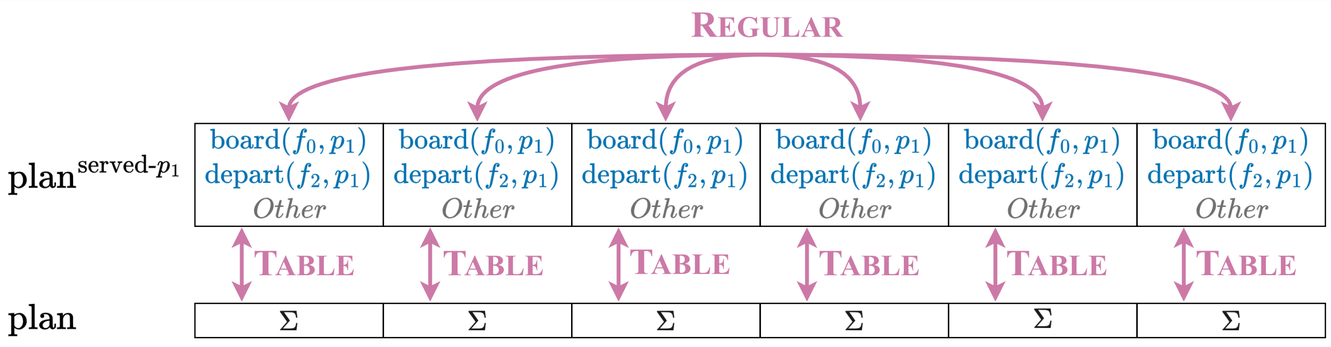






CP Model (BASE)

The plan is an array of variables with a fixed length. Each automaton then enforces its dynamics through **REGULAR** constraints on an individual plan that is then linked to the main plan with **TABLE** constraints.



Operator-Counting – Extending the CP model

Operator-counting constraints (OPC) are linear systems of inequalities that express knowledge about the number of occurrences of action x_a in all plans of a task.

Landmark Constraints (LMC)
"One of these actions must be present"

- $x_{depart(f_2,p_1)} \geq 1$
- $x_{board(f_0,p_1)} \geq 1$
- $x_{up(f_0,f_2)} + x_{up(f_1,f_2)} \geq 1$

Net Change Constraints (NCC)
"If an atom is in the goal but not the initial state, any plan must establish it once more than it deletes it."

- $x_{up(f_0,f_2)} + x_{up(f_1,f_2)} - x_{down(f_2,f_0)} - x_{down(f_2,f_1)} \geq 0$

Grouping actions with the same sign in each OPC reduces the **occurrence variables** to track off. After partition refinement, each variable is linked to the plan via **AMONG** constraints, counting the specified action(s):

$Occ_{\{depart(f_2,p_1)\}}$, $Occ_{\{board(f_0,p_1)\}}$, $Occ_{\{up(f_0,f_2),up(f_1,f_2)\}}$, $Occ_{\{down(f_2,f_0),down(f_2,f_1)\}}$, $Occ_{Remaining}$

The OPC are added to the model as the following **redundant constraints**:

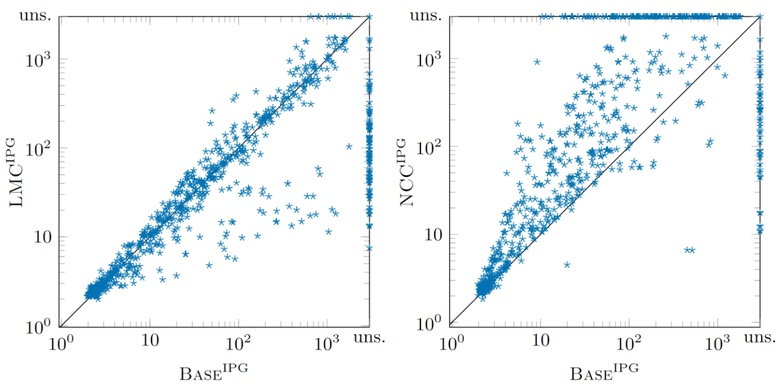
- $Occ_{\{depart(f_2,p_1)\}} \geq 1$
- $Occ_{\{board(f_0,p_1)\}} \geq 1$
- $Occ_{\{up(f_0,f_2),up(f_1,f_2)\}} \geq 1$
- $Occ_{\{up(f_0,f_2),up(f_1,f_2)\}} \geq Occ_{\{down(f_2,f_0),down(f_2,f_1)\}}$

Lastly, the **sum of all actions occurrences** is bound to the length of the plan with:

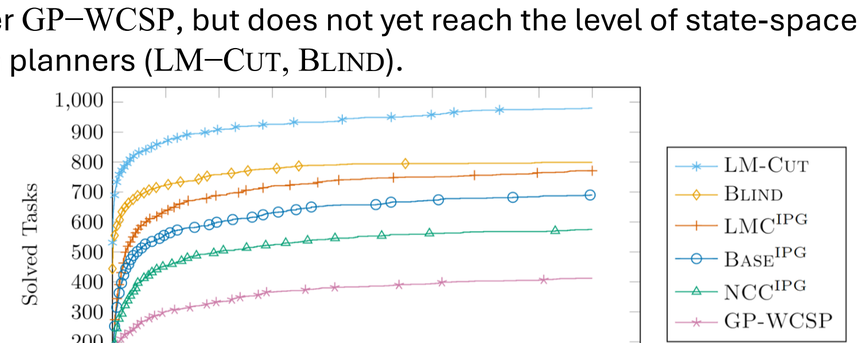
- $Occ_{\{depart(f_2,p_1)\}} + Occ_{\{board(f_0,p_1)\}} + Occ_{\{up(f_0,f_2),up(f_1,f_2)\}} + Occ_{\{down(f_2,f_0),down(f_2,f_1)\}} + Occ_{Remaining} = l$

Results

The **OPC brings valuable information** to the model. LMC solves the most instances. NCC performs better than BASE and LMC on some specific planning domains but has the disadvantage of adding more computational overhead on the runtime and runs out of memory more often.



The approach beats state-of-the-art CP-based optimal classical planner GP–WCSP, but does not yet reach the level of state-space search planners (LM–CUT, BLIND).



LINKÖPING UNIVERSITY
Machine Reasoning Lab
WASP WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM
POLYTECHNIQUE MONTRÉAL