Dresden University of Technology

Daniel Gnad, Markus Hecher, **Sarah Gaggl**, Dominik Rusovac, David Jakob Speck and Johannes K. Fichte

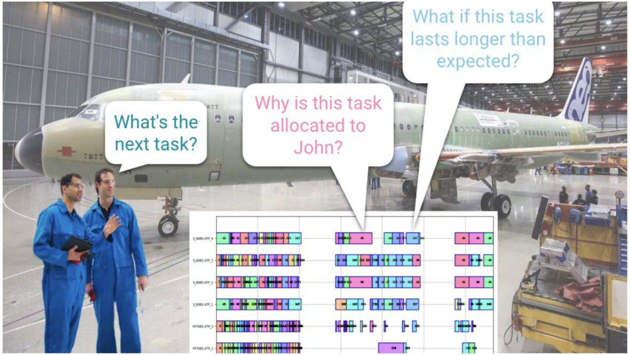
Interactive Exploration of Plan Spaces

KR@Melbourne, 16th November 2025

Motivation

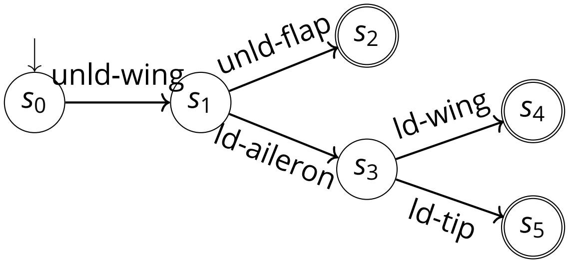
- Many planning applications require a set of plans
- Recently researchers started investigation of preferences, enumerate plans by top-k planning, or count plans to reason about the plan space
- Unfortunately: reasoning about plan space is computationally extremely hard
- Feeding many similar plans to user is hardly practical

Motivation ctd.



- Navigate plan spaces interactively to explore, understand and explain solutions
- Motivated by logistics application at Airbus SE - Beluga Competition

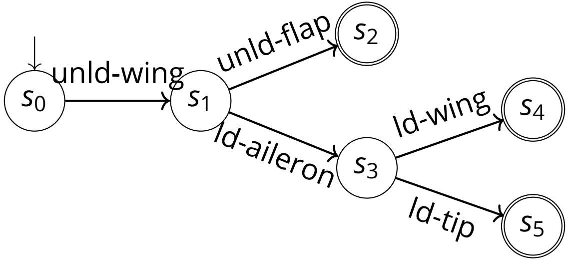
Example



3 valid plans:

1. unload-wing; unload-flap
2. unload-wing; load-aileron; load-wing
3. unload-wing; load-aileron; load-tip

Example



3 valid plans:

1. unload-wing; unload-flap
2. unload-wing; load-aileron; load-wing
3. unload-wing; load-aileron; load-tip

Faceted actions [Speck et al. 2025]

- Meaningful actions that can be used by some plan but not all plans

Agenda

Navigating with Facets

Navigating Plans with Facets

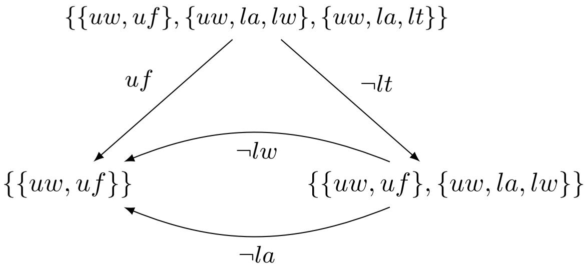
Facet of planning task

- Planning task: $\Pi = \langle \mathcal{A}, \mathcal{O}, \mathcal{I}, \mathcal{G} \rangle$
- $a@j$ and $\neg a@j$ facet of Π: if there is a witnessing plan where a is at the j-th position and some plan where a is not at j
- a or $\neg a$ facet of Π (without fixing position)

Routes over Π

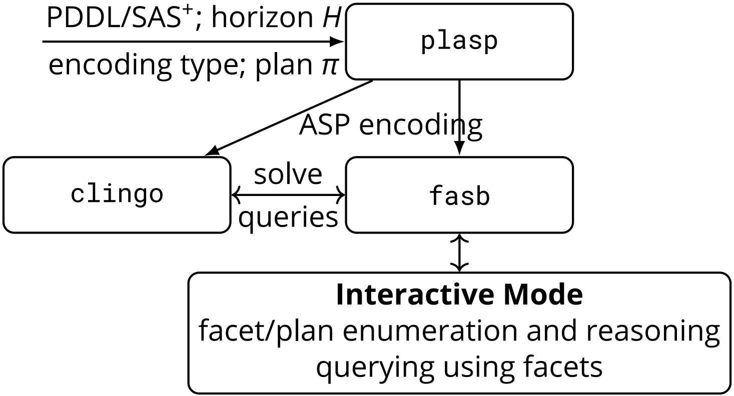
- Route δ: finite sequence $\langle f_1, \ldots, f_n \rangle$ of facets $f_i \in \mathcal{F}(\Pi)$
- Plans(Π^δ): set of plans for Π under δ

Navigating the Plan Space



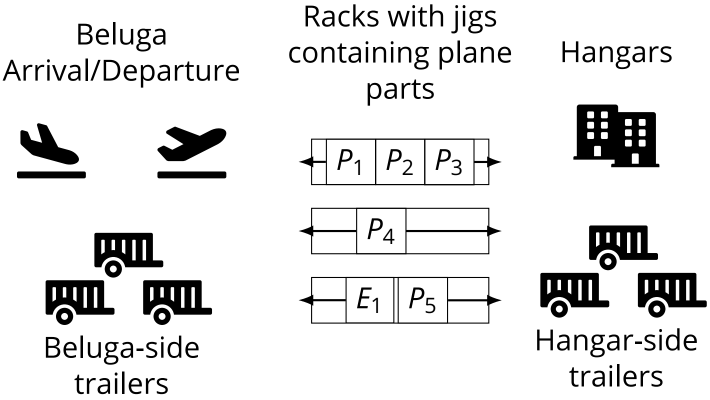
- $\mathcal{F}(\Pi) = \{uf, la, lt, lw, \neg uf, \neg la, \neg lt, \neg lw\}$
- $\delta_1 = \langle uf \rangle, \delta_2 = \langle \neg lt, \neg la \rangle, \delta_3 = \langle \neg lt, \neg lw \rangle$
- Plans$(\Pi^{\langle \neg lt \rangle}) = \{\{uw, uf\}, \{uw, la, lw\}\}$

PlanPilot

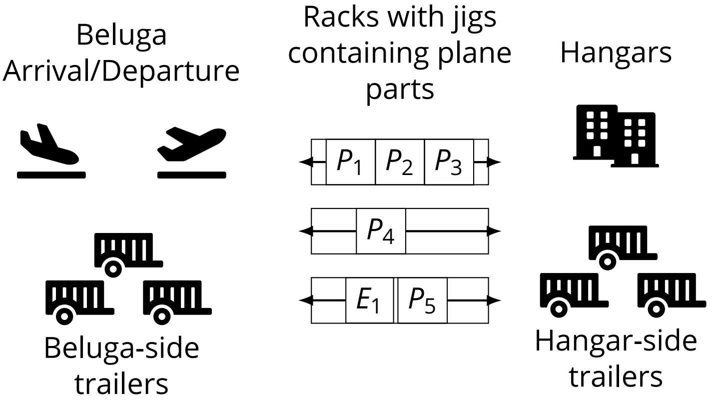

PDDL/SAS+; horizon H
encoding type; plan π
plasp
ASP encoding
clingo
solve
queries
fasb
Interactive Mode
facet/plan enumeration and reasoning
querying using facets

Navigating the Beluga Challenge

The Beluga Problem



The Beluga Problem



Many constraints

- Flight schedule changes/Maintenance: keep one rack empty (robustness)
- Soft constraints: smaller jigs on smaller racks; empty jigs on Beluga side; full jigs on factory side

Beluga Explanation Challenge

- Tight space constraints at assembly facilities and limited mobility of jigs
- Plans obtained from automated planning system are unlikely to be robust enough to ensure executability under uncertain flight schedules
- Fully representing all constraints in the planning model is not feasible

Beluga Explanation Challenge

- Tight space constraints at assembly facilities and limited mobility of jigs
- Plans obtained from automated planning system are unlikely to be robust enough to ensure executability under uncertain flight schedules
- Fully representing all constraints in the planning model is not feasible

Practical approach

Generate solution with planner and ask for explanations for certain decisions (actions taken, implications for alternatives) and possibly refine the plan.

Queries Adapted from Challenge

Q1: Why is jig X loaded on rack A instead of another rack B?

Q2: Why load jig B on rack D instead of loading jig C on rack A?

Q3: Why not load jig C on rack A before loading jig B on rack D?

Queries Adapted from Challenge

Q1: Why is jig X loaded on rack A instead of another rack B?

Q2: Why load jig B on rack D instead of loading jig C on rack A?

Q3: Why not load jig C on rack A before loading jig B on rack D?

Feasibility of Alternative Solutions

Along with the query, we get a concrete plan $\pi = \langle a_1, \ldots, a_n \rangle$ that allows us to identify the action(s) in question

- Enforce plan prefix $\pi_p = \langle a_1, \ldots, a_{i-1} \rangle$ up to (excluding) the first action a_i in question by navigating along the route $\delta_p = \langle a_1@1, \ldots, a_{i-1}@i-1 \rangle$
- Then, we check if the alternative action a' for step i is a facet, i.e., if $\delta_p^i = \delta_p \circ a'@i$ is a route
- To analyze resulting plan space: count number of plans $|Plans(\Pi^{\delta_p^i})|$
- If alternative action or order is not feasible: increase horizon

Queries Adapted from Challenge ctd.

Q4 How can we reduce the number of swaps?

Queries Adapted from Challenge ctd.

Q4 How can we reduce the number of swaps?

Minimizing the Number of Swaps

- Swaps are special sequences of actions: take jig out of rack and either put it back, or put it in other rack
- Swaps add redundant actions to plans
- They can be detected by searching for specific actions: taking full jig out at Beluga side/ taking empty jig out at hangar side
- Denote set of such actions by $\mathcal{O}^{\text{swap}}$
- In original plan π, we can either fix the actions up to the first swap action, or start from scratch without keeping actions from π
- Navigate plan space with navigation steps $\neg a$ for actions $a \in \mathcal{O}^{\text{swap}}$
- Once a swap action becomes cautious, i.e., $\mathcal{CC}_{\ell}(\Pi) \cap \mathcal{O}^{\text{swap}} \neq \emptyset$, we know that a swap is required for the current route

Queries Adapted from Challenge ctd.

Q5 What is the impact of removing rack A for maintenance?

Q6 How can we keep one rack empty all the time?

Queries Adapted from Challenge ctd.

Q5 What is the impact of removing rack A for maintenance?

Q6 How can we keep one rack empty all the time?

Handling Rack Removal

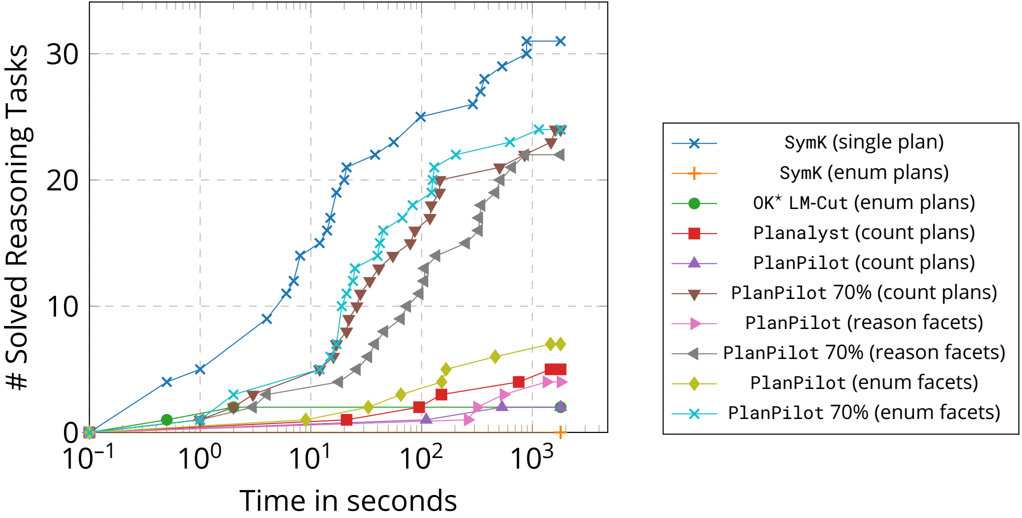
- Identify actions that put some jig into one of the respective racks
- Forbid these actions using facets
- In every navigation step: check if such an action becomes a cautious consequence
- If this is the case: at least one action is needed that puts a jig into that rack
- Can answer question on impact of removing rack
- More flexible than: removing rack from task description and checking if a plan exists

Experiments

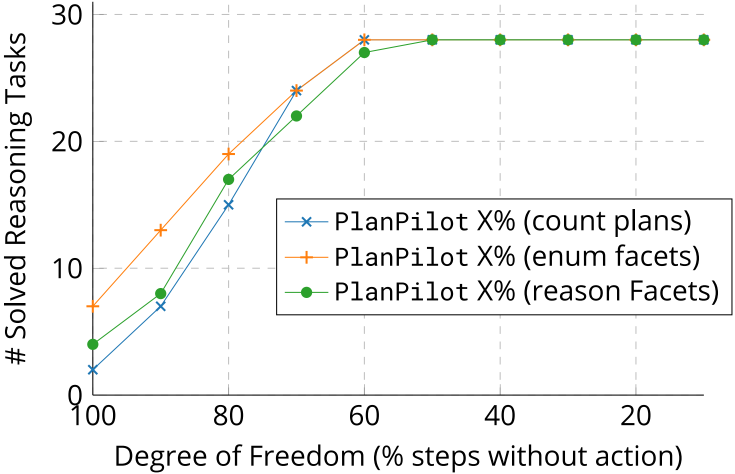
Experimental Setting

- Considered the 48 solvable instances of Beluga explainability challenges
- We tested interaction and explainability based on a given plan
- Used optimal planner to determine an optimal plan for each instance (`SymK` planner [Speck, Seipp, and Torralba 2025])
- Optimal plan for 31 instances
- Limits: memory 3.5 GB, time 30 min per instance
- To test `PlanPilot`: invoked it with a query to count plans, enumerate facets, and report significance of facets.

Results for solving or reasoning over given number of instance

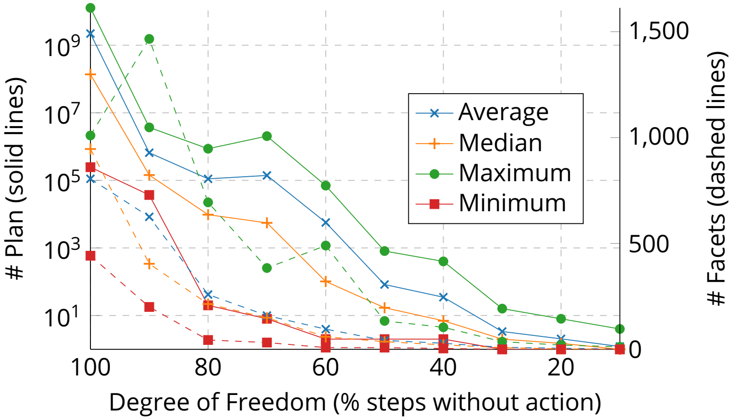


Results with Different Degrees of Freedom



Number of instances (y-axis) for which `PlanPilot` was able to count plans, list facets, or reason about the facet significance

Results with Different Degrees of Freedom ctd.



The number of plans (left y-axis) and facets (right y-axis) as determined by our experiments.

Conclusion

Conclusion

Main Contributions

- Concrete use case to comprehend large solution spaces and explore alternative solutions - 2025 Beluga AI Challenge
- Practical approach to navigate plan spaces iteratively and interactively
- `PlanPilot` allows to output, filter, count plans, and restrict flexibility in plans or count flexibility in actions or estimate effects

Conclusion

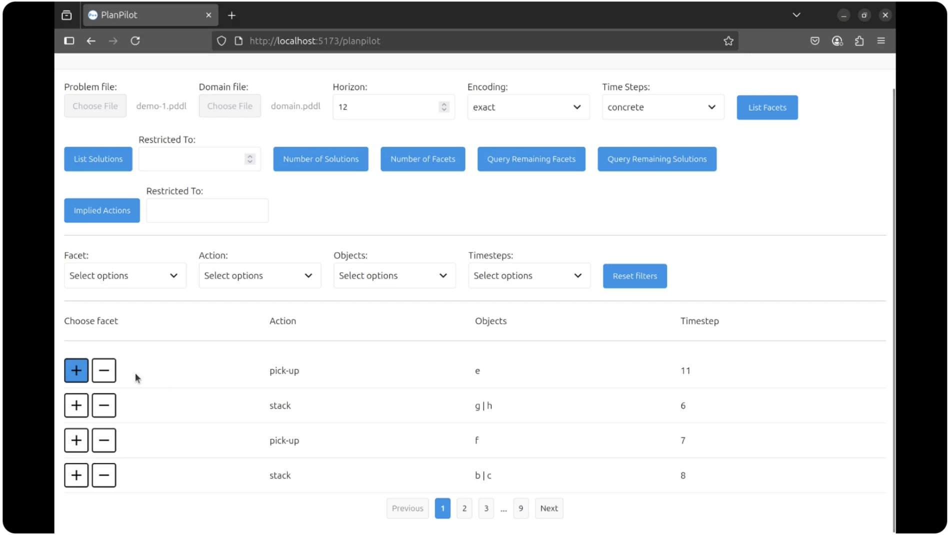
Main Contributions

- Concrete use case to comprehend large solution spaces and explore alternative solutions - 2025 Beluga AI Challenge
- Practical approach to navigate plan spaces iteratively and interactively
- `PlanPilot` allows to output, filter, count plans, and restrict flexibility in plans or count flexibility in actions or estimate effects

Future Work

- Obtaining (minimal) partially ordered plans natively without adding implications to simulate all steps
- Loopless plans
- Evaluating our tool on a larger set of practical instances

Graphical PlanPilot



ICAPS 2025 Demo PlanPilot

Beluga Challenge Winners

Linköping University team: Elliot Gestrin, Gustaf Söderholm, Paul Höft, Mauricio Salerno, Jendrik Seipp, and led by Daniel Gnad won the Explainability Challenge