Satisficing and Optimal Generalised Planning via Goal Regression

LAAS CNRS
RWTH AACHEN UNIVERSITY


Dillon Z. Chen


Till Hofmann


Toryn Q. Klassen


Sheila A. McIlraith

VECTOR INSTITUTE | INSTITUT VECTEUR
UNIVERSITY OF TORONTO

PDDL (STRIPS) Planning

Sequential decision making problems over formally defined models

A **domain** is a set of first-order predicates and action schemata $\mathcal{D} = \langle \mathcal{P}, \mathcal{A} \rangle$

A **problem** is a domain, initial state, goal cond. and finite set of objects $P = \langle \mathcal{D}, s^0, g, O \rangle$

A **plan** α is sequence of actions that progresses s^0 to a state satisfying g

PDDL models exhibit rich *structure* to help generate solutions efficiently

PDDL Planning: Household Robot Example

Domain

```
(:action move
   :parameters  (?from ?to)
   :precondition (and (atRobot ?from))
   :effect (and  (atRobot ?to)
      (not (atRobot ?from))))
```

action schema

```
(:action pickUp
   :parameters (?obj ?loc)
   :precondition  (and (at ?obj ?loc)
      (atRobot ?loc) (handFree))
   :effect (and (holding ?obj) (not (at ?obj ?loc))
      (not (handFree))))
```

predicate

```
(:action putDown
   :parameters (?obj ?loc)
   :precondition (and (holding ?obj) (atRobot ?loc))
   :effect (and (at ?obj ?loc) (handFree)
      (not (holding ?obj))))

...
```

Problem

```
(:objects dog ball apple mango cake)
```

objects

```
(:init
   (hungry dog)
   (at mango bedroom)
   (at cake livingRoom)
   (at apple kitchen)
   (at ball backyard)
   (atRobot backyard)
)
```

initial state

```
(:goal
   (at cake kitchen)
   (at ball storageRoom)
)
```

goal condition

Problem Statement: Generalised Planning (GP)

Generalised planning problem:

- a domain $\mathcal{D}$
- training planning problems $\mathcal{P}_{train}$ from $\mathcal{D}$
- testing planning problems $\mathcal{P}_{test}$ from $\mathcal{D}$

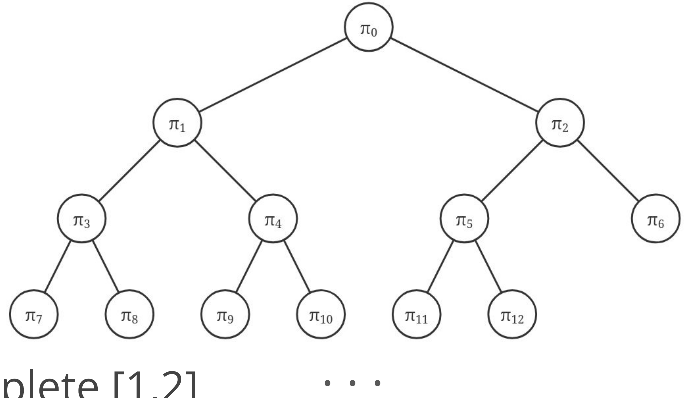
Generalised plan: is a *program* π that

- is *synthesised* from $\mathcal{P}_{train}$
- can be *instantiated* to solve problems in $\mathcal{P}_{test}$

generalisation in sequential decision-making problems across:

- unseen states
- unseen goals
- arbitrary number of objects

Generalised Planning is Hard and Interesting!



- Space of generalised plans is huge
- Some models of GP problems are EXPSPACE-complete [1,2]
- Generalisation is difficult and a core problem of AI

[1] Siddharth Srivastava, Shlomo Zilberstein, Neil Immerman, Hector Geffner: Qualitative Numeric Planning. AAAI 2011
[2] Blai Bonet, Hector Geffner: Qualitative Numeric Planning: Reductions and Complexity. J. Artif. Intell. Res. 69: 923-961 (2020)

Current Approaches

- Deep/machine learning approaches
  - imitation learning
  - e.g. GNNs, Transformers
  - ✔ fast to synthesise policies
  - ✗ low expressivity, not interpretable
- Symbolic/abstraction approaches
  - theorem proving
  - e.g. QNP, ASP
  - ✗ slow to synthesise policies
  - ✔ expressive, interpretable

Our approach: Goal Regression

Goal regression [3, 4] computes the minimal and sufficient condition for achieving a goal ***g*** via an action ***a***

⇒ ✔ efficient policy synthesis

⇒ ✔ expressive and interpretable policies

- PDDL STRIPS goal regression is defined by

$$regr(g, a) = (g \setminus add(a)) \cup pre(a)$$

[3] Richard Fikes, Peter E. Hart, Nils J. Nilsson: Learning and Executing Generalized Robot Plans. Artif. Intell. 3(1-3): 251-288 (1972)
[4] Raymond Reiter: The Frame Problem in the Situation Calculus: A Simple Solution (Sometimes) and a Completeness Result for Goal Regression. Artificial and Mathematical Theory of Computation 1991

Methodology: (1) Synthesising GenPlans via Goal Regression

Synthesise a GPlan $\boldsymbol{\pi}$ in the form of a set of first-order rules from $\mathcal{P}_{train}$ by

1. compute optimal plans $\{a_1, \ldots, a_n\}$ for single goal atoms in some order $\{g_1, \ldots, g_n\}$ for each training problem $P \in \mathcal{P}_{train}$
2. perform goal regression on goals g_i with corresponding plans π_i to get a set of partial-state, macro-action pairs $\langle \sigma_i, a_i \rangle$ where $a_i = \alpha_1, \ldots, \alpha_q$
3. lift the set of pairs $\langle \sigma_i, a_i \rangle$ and goals g_i into a set of first-order rules

state condition / goal condition / actions

$$\left\{ \exists \{X\} \; \bigwedge_{i=1,\ldots,m} p_i^s(X_i^s) \wedge \bigwedge_{j=1,\ldots,n} p_j^g(X_j^g) \rightarrow \alpha_1(X_1^a), \ldots, \alpha_q(X_q^a) \right\}$$

transportation domain

STRIPS Domain

```
putDown
 var: ?obj, ?loc
 pre: atRobot(?loc), holding(?obj)
 add: at(?obj, ?loc), handFree()
 del: holding(?obj)

move
 var: ?from, ?to
 pre: atRobot(?from)
 add: atRobot(to)
 del: atRobot(?from)

...
```

STRIPS Domain

```
putDown
 var: ?obj, ?loc
 pre: atRobot(?loc), holding(?obj)
 add: at(?obj, ?loc), handFree()
 del: holding(?obj)
```

```
move
 var: ?from, ?to
 pre: atRobot(?from)
 add: atRobot(to)
 del: atRobot(?from)
```

...

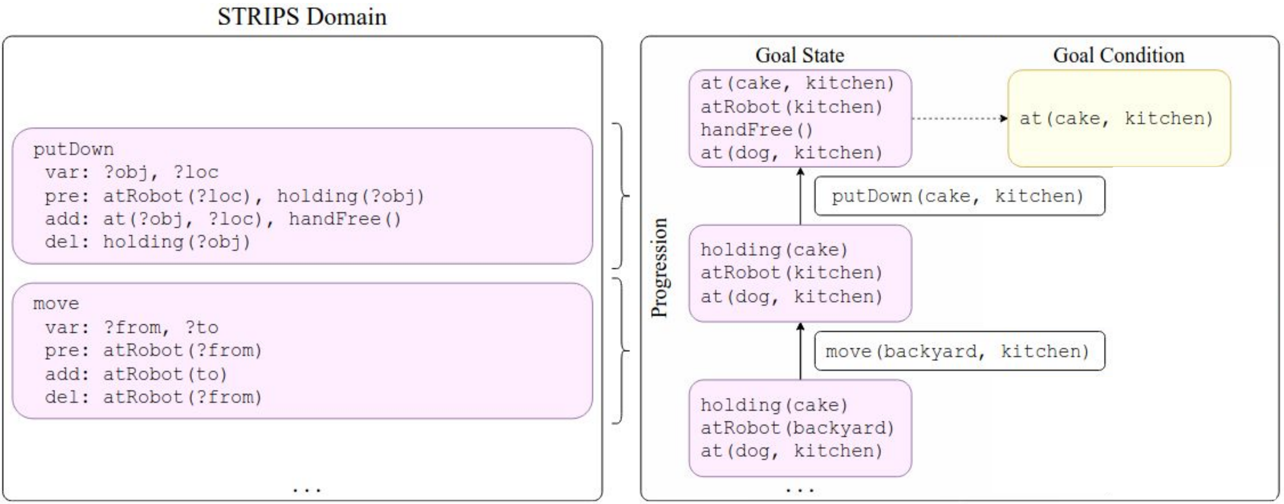
Goal Condition

```
at(cake, kitchen)
```

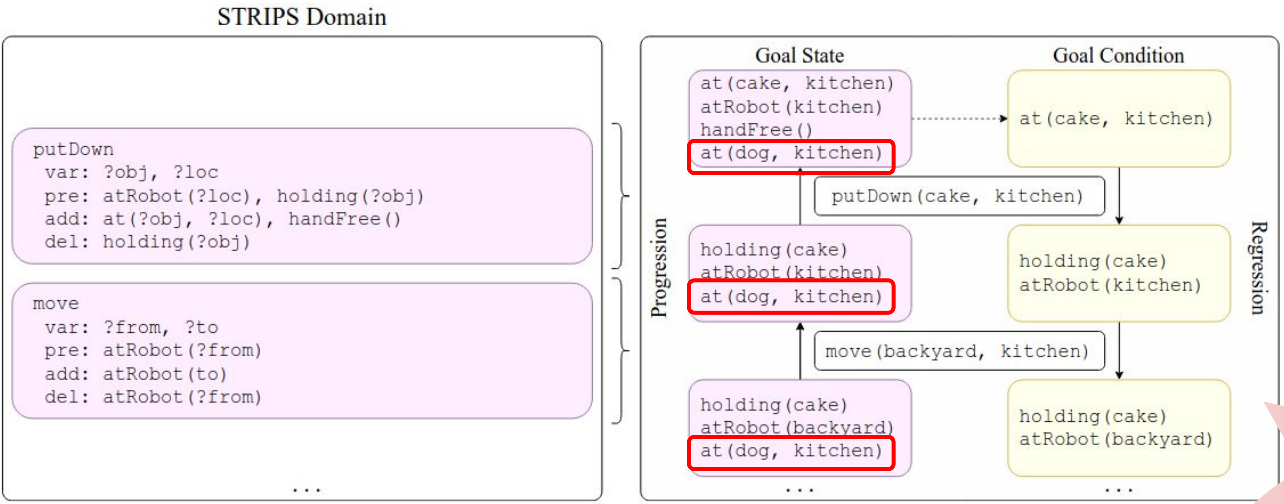
```
holding(cake)
atRobot(backyard)
at(dog, kitchen)
```

...

initial state
and
goal condition

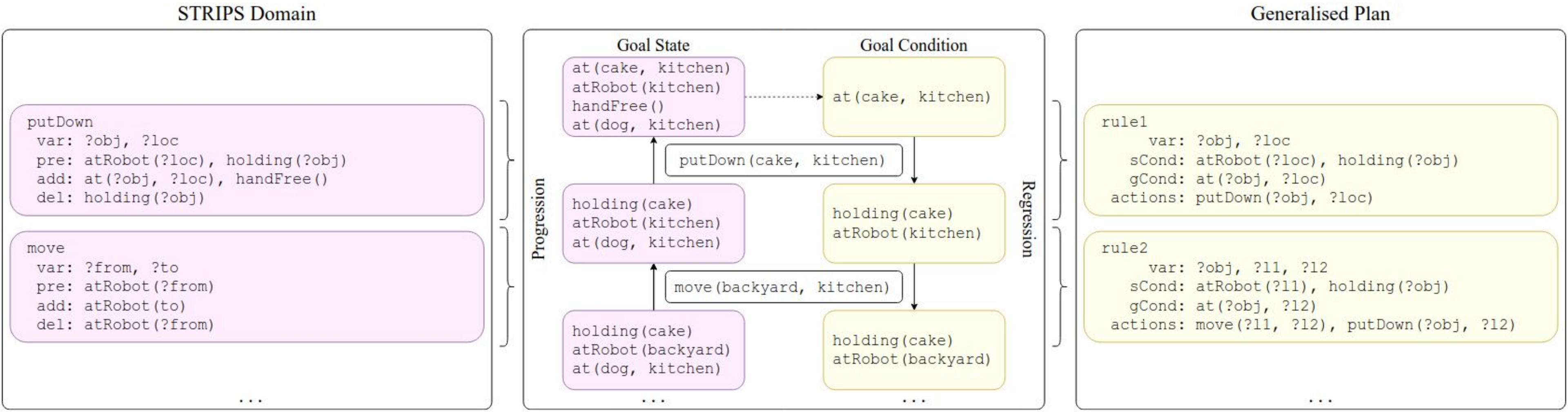


find a plan and progress the initial state



regress the goal with the plan

regression deems `at(dog, kitchen)` irrelevant lift the regressed states into rules



Methodology: (2) Instantiating GenPlans via Database Algorithms

speed focused GP

Instantiate a GPlan $\boldsymbol{\pi}$ on a problem $\boldsymbol{P} \in \mathcal{P}_{test}$ by treating it as a policy

1. set $\boldsymbol{s} = \boldsymbol{s_0}$ and while the goal has not been achieved, repeat the following steps
2. ground a lifted rule where $\wedge_{i=1,\ldots,m} p_i^s(X_i^s)$ holds in $\boldsymbol{s}$ and $\wedge_{j=1,\ldots,n} p_j^g(X_j^g)$ holds in $\boldsymbol{g} \setminus \boldsymbol{s}$
3. apply corresponding sequence of actions $\alpha_1(X_1^a), \ldots, \alpha_q(X_q^a)$ on $\boldsymbol{s}$

ground with first-order query algorithms

⇒ MOOSE Generalised Planner

1. *Goal regression* for generalised plan synthesis
2. *Database algorithms* for policy execution
3. …

Experimental Results

Benchmarks: HUGE numbers of objects

	Max Training #Objects	Max Testing #Objects
Barman	27	853
Ferry	8	1461
Gripper	5	48500
Logistics	29	1260
Miconic	11	1950
Rovers	36	596
Satellite	43	402
Transport	17	354

Synthesis Experiments

- Compare against *3 configurations* of the Sketch Learner [5] generalised planner
- 32 GB memory
- 12 hour runtime limit
- 5 repeats per domain

[5] Dominik Drexler, Jendrik Seipp, Hector Geffner: Learning Sketches for Decomposing Planning Problems into Subproblems of Bounded Width. ICAPS 2022

Synthesis Results

Average time and memory usage (↓)

MOOSE uses <1GB memory and synthesises GenPlans for all domains

	Time (s)				Memory (MB)			
	SLEARN-0	SLEARN-1	SLEARN-2	MOOSE	SLEARN-0	SLEARN-1	SLEARN-2	MOOSE
Barman	-	-	-	**202**	-	-	-	**184**
Ferry	21	12	**2**	9	184	134	76	**52**
Gripper	**3**	9	45	10	66	142	391	**64**
Logistics	-	-	-	**71**	-	-	-	**73**
Miconic	57	**1**	3	12	381	56	125	**52**
Rovers	-	-	-	**534**	-	-	-	**187**
Satellite	-	-	1559	**514**	-	-	7598	**82**
Transport	-	**12**	**12**	21	-	114	129	**80**

Satisficing Planning Experiments

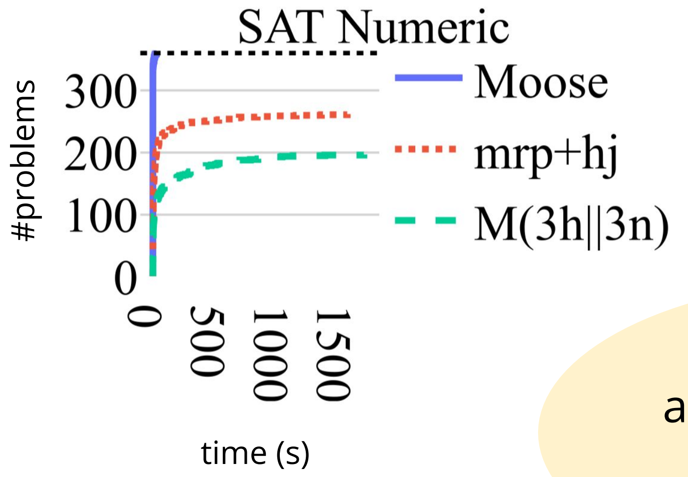
- 8 classical domains and 4 numeric domains
- Compare against:
  - Classical planners: Sketch Learner [5], LAMA [6]
  - Numeric planners: ENHSP(mrp+hj) [7], ENHSP(M(3h||3n) [8]
- 8 GB memory
- 30 minute runtime limit
- 5 repeats per problem

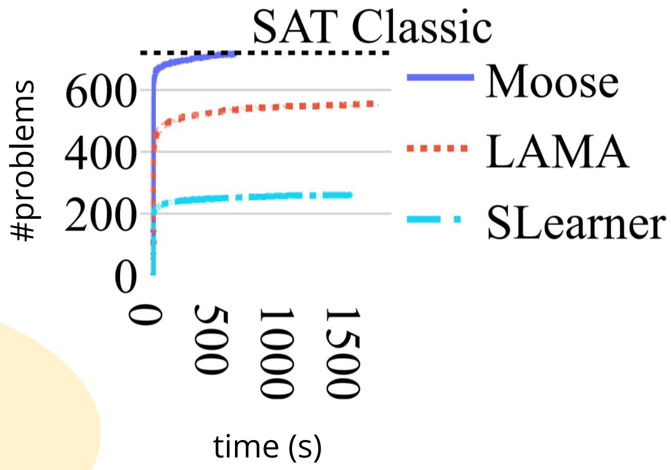
[5] Dominik Drexler, Jendrik Seipp, Hector Geffner: Learning Sketches for Decomposing Planning Problems into Subproblems of Bounded Width. ICAPS 2022
[6] Silvia Richter, Matthias Westphal: The LAMA Planner: Guiding Cost-Based Anytime Planning with Landmarks. J. Artif. Intell. Res. 39: 127-177 (2010)
[7] Enrico Scala, Alessandro Saetti, Ivan Serina, Alfonso Emilio Gerevini: Search-Guidance Mechanisms for Numeric Planning Through Subgoaling Relaxation. ICAPS 2020
[8] Dillon Z. Chen, Sylvie Thiébaux: Novelty Heuristics, Multi-Queue Search, and Portfolios for Numeric Planning. SOCS 2024

Satisficing Planning Results

Cumulative coverage (↑)

The number of problems (*y-axis*) that a planner solves within *n* seconds (*x-axis*)





MOOSE solves almost all problems much faster than the baselines

We're not done yet!

Methodology: (3) Instantiating GenPlans via Search

Instantiate a GPlan $\boldsymbol{\pi}$ on a problem $\boldsymbol{P} \in \mathcal{P}_{test}$ with search space pruning via PDDL axioms

quality focused GP

1. encode axioms that detect unachieved goals

$$p_{ug}(X) \text{ :- } p_g(X) \wedge \neg p(X)$$

2. encode axioms that restrict action application based on learned rules

$$(\alpha_1)_\pi(X) \text{ :- } \bigwedge_{i=1,\dots,m} p_i^s(X_i^s) \wedge \bigwedge_{j=1,\dots,n} (p_j^g)_{ug}(X_i^g)$$

3. feed transformed PDDL problem into a planner that supports axioms

Returns optimal plans under certain conditions

⇒ MOOSE Generalised Planner

1. *Goal regression* for generalised plan synthesis
2. *Database algorithms* for policy execution
3. *PDDL axioms* for search space pruning

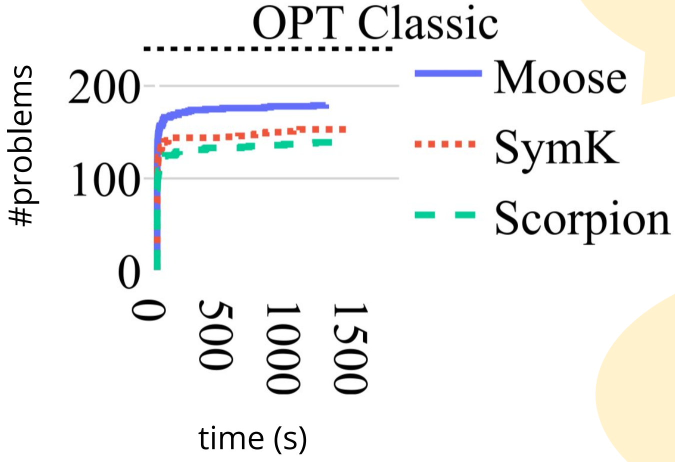
Optimal Planning Experiments

- 8 classical domains
- use SymK [9] as downstream planner that supports PDDL axioms
- Compare against SymK without axioms and Scorpion [10]
- 8 GB memory
- 30 minute runtime limit
- 5 repeats per problem

[9] David Speck, Jendrik Seipp, Álvaro Torralba: Symbolic Search for Cost-Optimal Planning with Expressive Model Extensions. J. Artif. Intell. Res. 82 (2025)
[10] Jendrik Seipp, Thomas Keller, Malte Helmert: Saturated Cost Partitioning for Optimal Classical Planning. J. Artif. Intell. Res. 67: 129-167 (2020)

Optimal Planning Results

Cumulative coverage (↑)



MOOSE solves more problems optimally in total

Coverage table by domain (↑)

Domain	SCORPION	SYMK	MOOSE
Barman	0	12	**24.6**
Ferry	17	18	**30.0**
Gripper	7	**30**	27.0
Logistics	**22**	10	15.0
Miconic	**30**	**30**	**30.0**
Rovers	18	**20**	**20.0**
Satellite	**26**	21	21.4
Transport	**20**	13	15.0
$\sum$ (240)	140	154	**183.0**

MOOSE usually improves upon its base planner (SymK)

Limitations and Future Work

- **Problem**: non-decomposable goals
  - **Solution**: learn goal orderings
- **Problem**: path-finding
  - **Solution**: transitive closure features, or search
- **Problem**: a posteriori policy termination
  - **Solution**: Sieve algorithm

Summary

Goal regression elicits powerful generalisation over structured models

Problem

Synthesise generalised plans for solving families of planning problems

$$regr(g, a) = (g \setminus add(a)) \cup pre(a)$$

Method

Synthesise via goal regression
→ improve synthesis efficiency
Instantiate via database query algorithms
→ improve planning speed
Instantiate via encoding rules as pruning axioms
→ improve solution quality

$$\{\exists\{X\} \underbrace{\bigwedge_{i=1,\dots,m} p_i^s(X_i^s)}_{\text{state condition}} \land \underbrace{\bigwedge_{j=1,\dots,n} p_j^g(X_j^g)}_{\text{goal condition}} \rightarrow \underbrace{a_1(X_1^a), \dots, a_q(X_q^a)}_{\text{actions}}\}$$

$$(a_1)_\pi(X) :- \bigwedge_{i=1,\dots,m} p_i^s(X_i^s) \land \bigwedge_{j=1,\dots,n} (p_j^g)_{ug}(X_i^g)$$

Theory

See paper for soundness and completeness theorems

	Time (s)				Memory (MB)			
	SLEARN-0	SLEARN-1	SLEARN-2	MOOSE	SLEARN-0	SLEARN-1	SLEARN-2	MOOSE
Barman	-	-	-	**202**	-	-	-	**184**
Ferry	21	12	**2**	9	184	134	76	**52**
Gripper	**3**	9	45	10	66	142	391	**64**
Logistics	-	-	-	**71**	-	-	-	**73**
Miconic	57	**1**	3	12	381	56	125	**52**
Rovers	-	-	-	**534**	-	-	-	**187**
Satellite	-	-	1559	**514**	-	-	7598	**82**
Transport	-	**12**	**12**	21	-	114	129	**80**

Experiments

Improvements on the 3 metrics of synthesis cost, instantiation cost, and solution quality