Endomorphic metalanguage and abstract planning for real-time intent recognition

Antoine Gréa

Lyon 1

12020-01-30T14:00+01
ISO-HE

- Directors
  - Samir Aknine
  - Lætitia Matignon
- Jury
  - Hamamache Kheddouci
  - Ivan Varzinczac

LIRIS

UNIVERSITÉ D'ARTOIS

- Reviewers
  - Eva Onainda
  - Damien Pellier

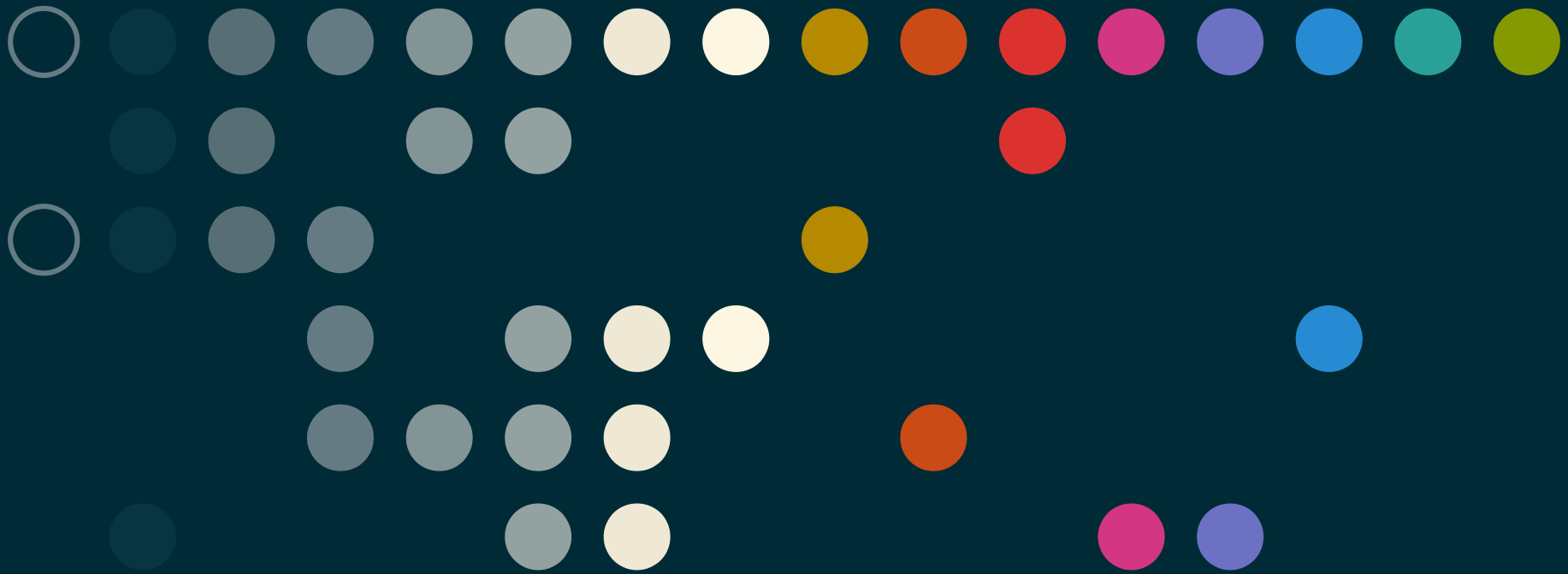
UNIVERSITAT POLITÈCNICA DE VALÈNCIA

LIG

Endomorphic metalanguage and abstract planning for real-time intent recognition

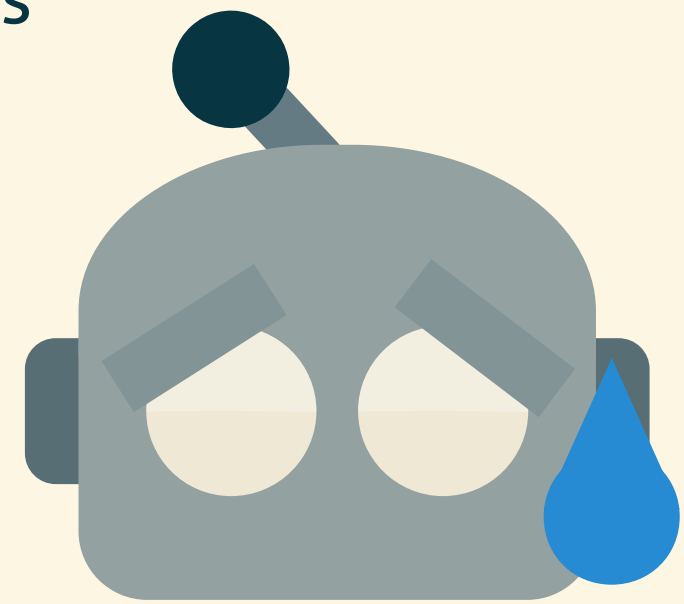
LIRIS Lyon 1 Antoine Gréa

1 Introduction



A what ?

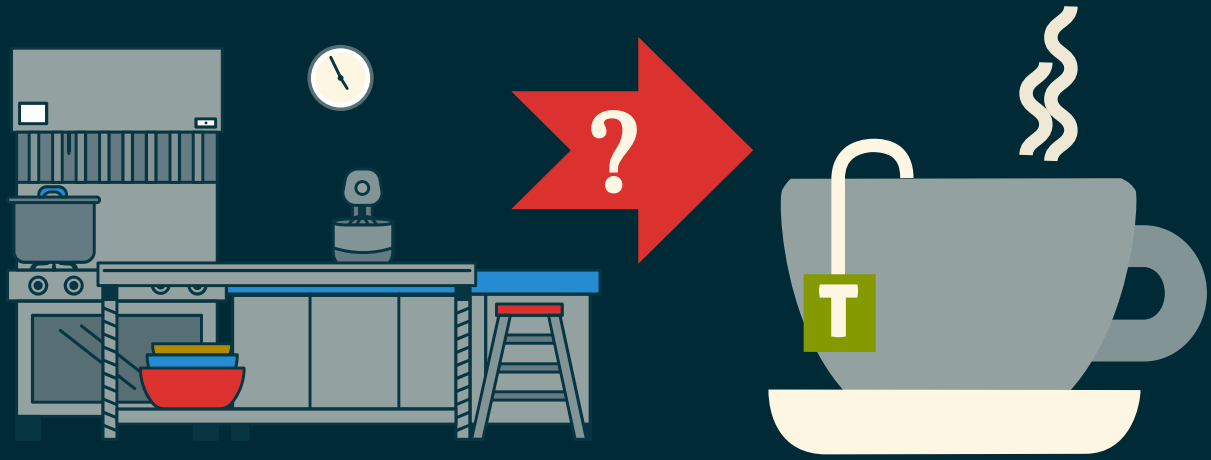
- *Dependent people need help !*
  - Not **annoying** the person
  - Can't see *everything* they are doing
- How to help without asking ?
  - Guessing the intent somehow
- **Intent recognition**
  - Observed behavior → Goal
  - Using action sequences: Plans



Kitchen Example

- Observation
  - Bob goes in the kitchen
- Available goals
  - Bob cleans the dishes
  - Bob makes tea
  - …
- **Infer correct one**

- Issues
  - Multiple goals
  - Interleaving
  - Partial Observation



Plan

1 Introduction

2 Intent Recognition

3 Knowledge Representation

4 General Planning

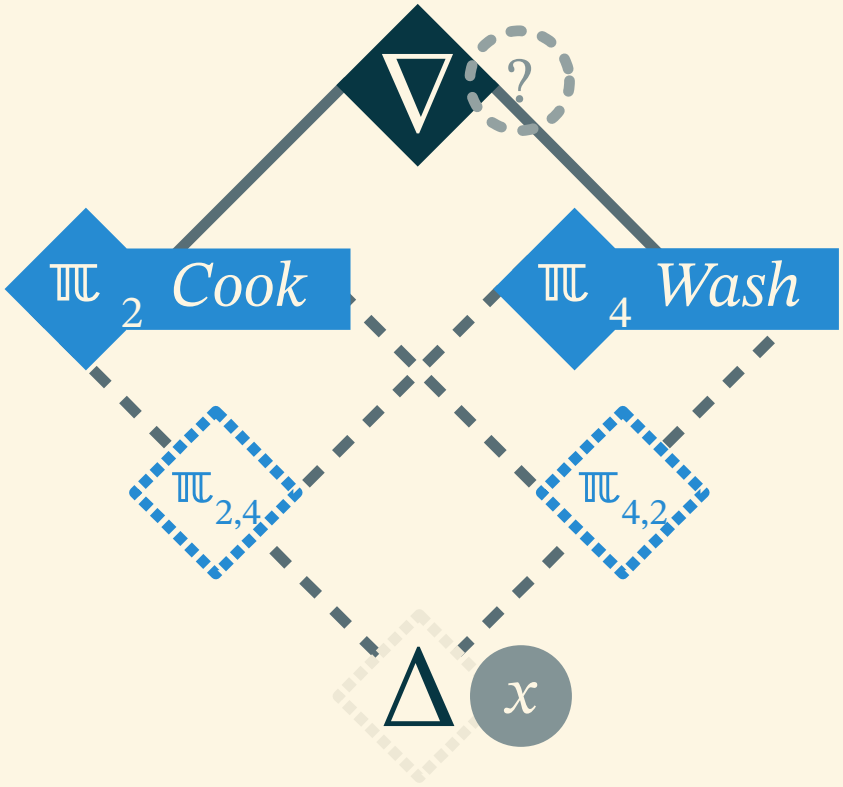
5 Flexible Online Planning

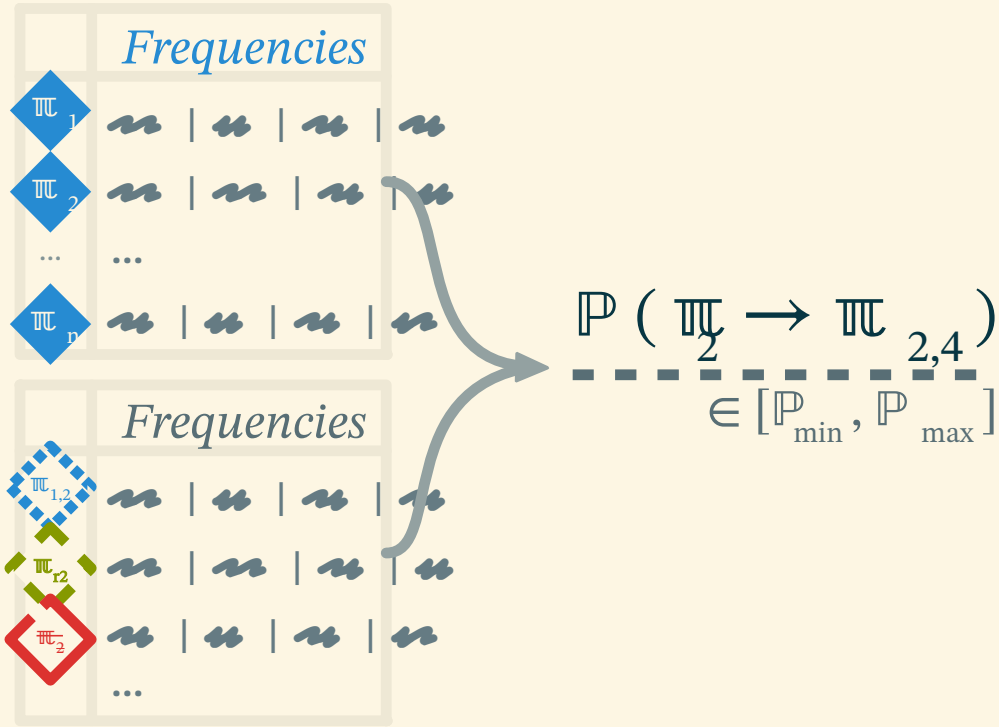
6 Conclusion

2 Intent Recognition

2.1 Logic Approach

- Lattice Based : ✔ Fast computations ✘ Exponential growth

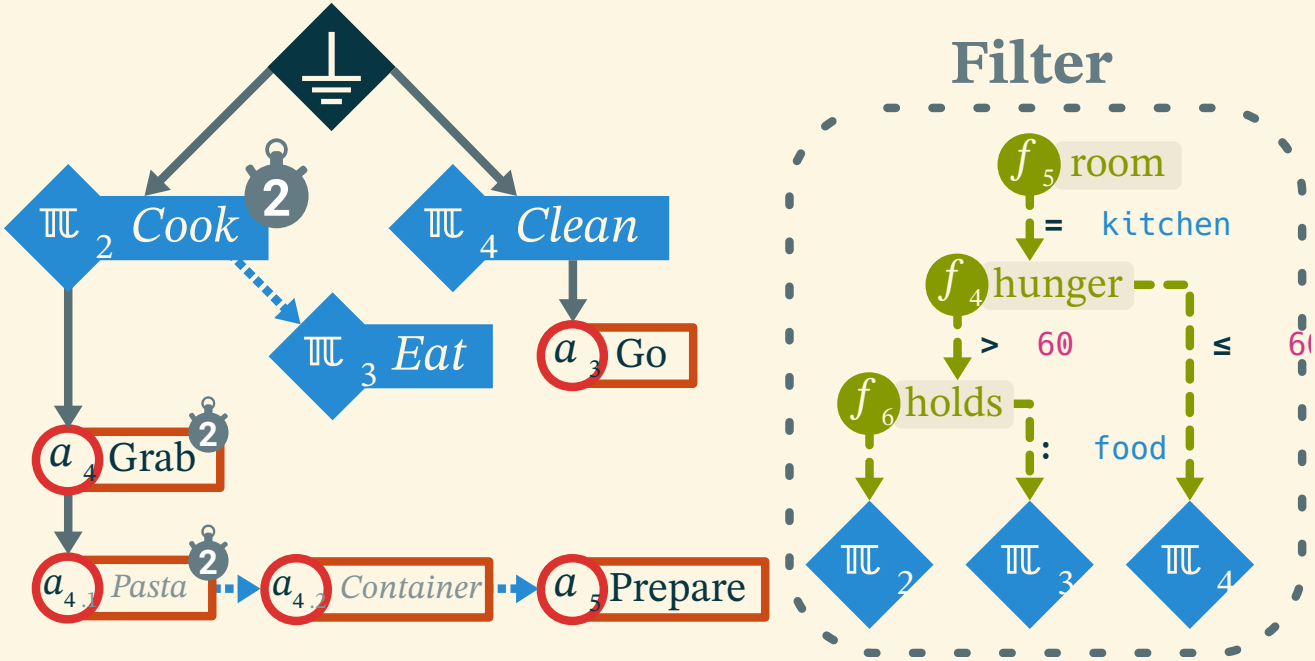




[@BOUCHARD_2006]

2.2 Stochastic Approach

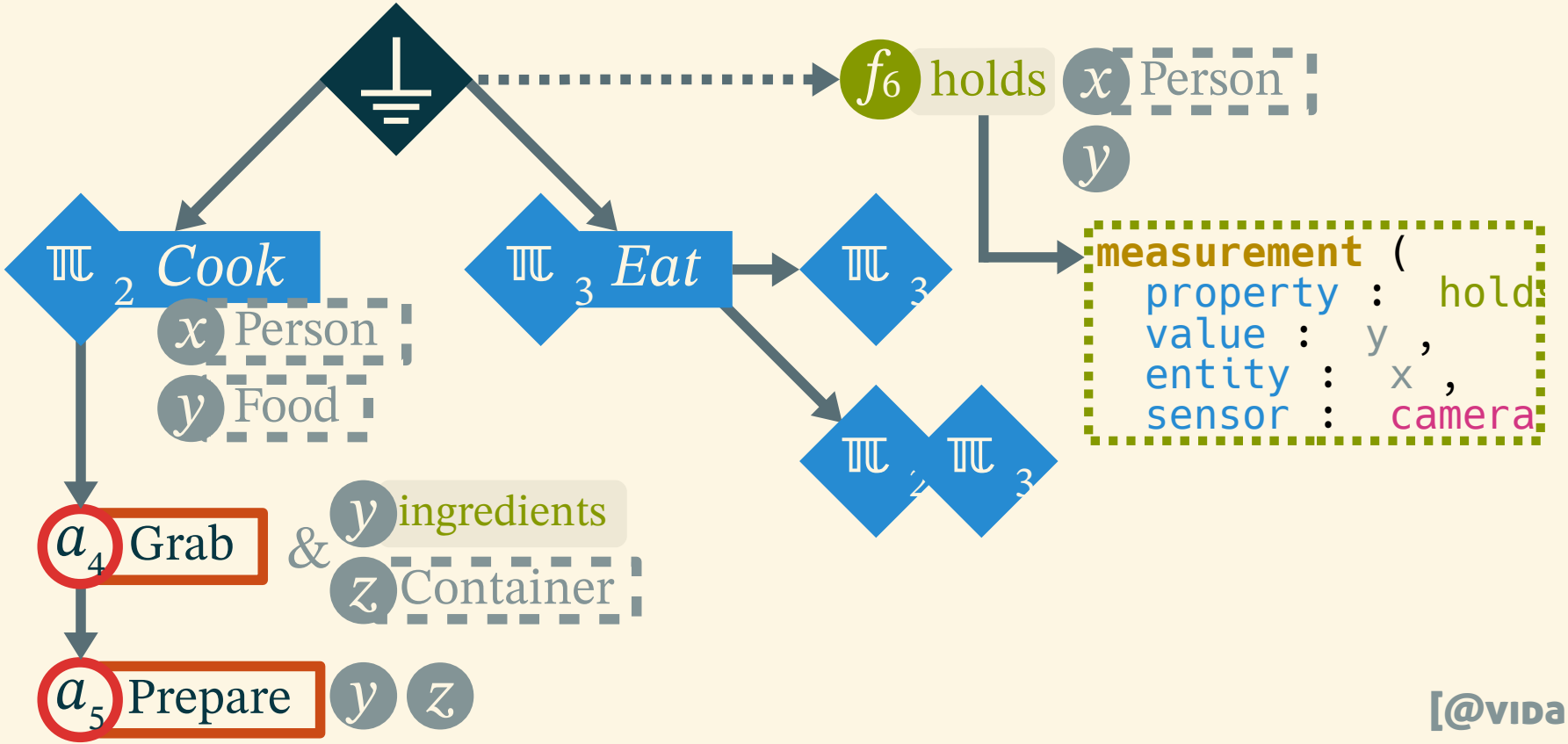
- And/Or and decision tree :
  - ✔ Accurate and efficient
  - ✘ Handmade plan library and tree



[@avrahami_2006]

2.3 Grammatical Approach

- Valued Grammar : ✔ Versatile ✘ Slow refresh rate (~40s)

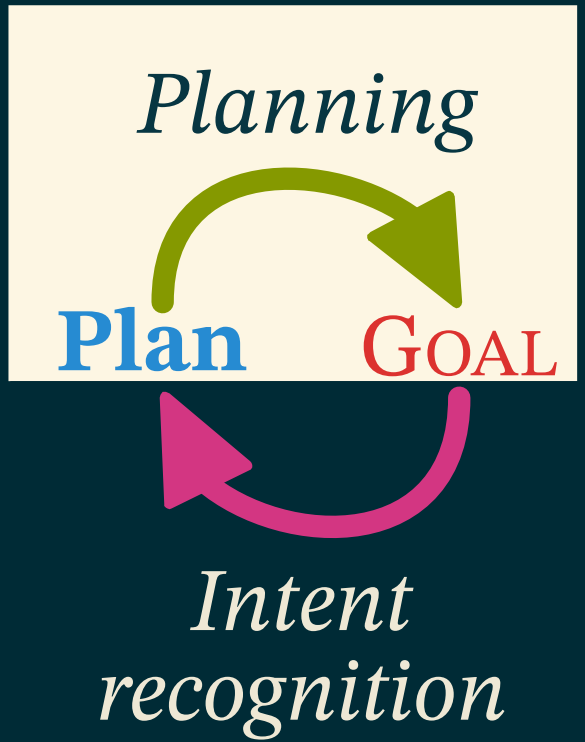


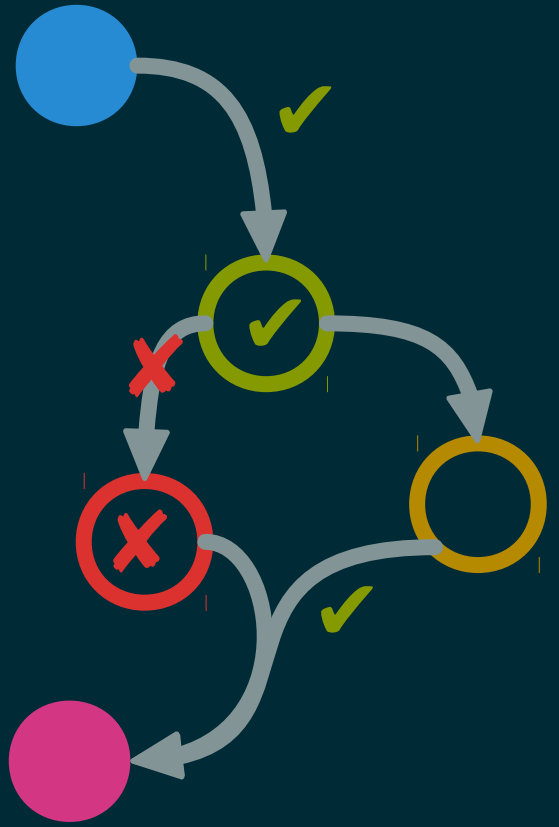
[@VIDAL_2010]

2.4 Invert Planning

- **Theory of Mind :**
  - ✔ Flexible
  - ✘ More complex

"*The easier the plan, the more likely the goal*





[@Ramirez_2008]

2.5 Framework Stacks

- Existing
- Contributions

3 Knowledge Representation

How to Know

- Abstraction
  - *How to **refer** to something*
- Formalization
  - *How to **talk** about something*
- Interpretation
  - *How to **know** about something*

Existing Tools

- Ontologies
  - Based on Description Logic

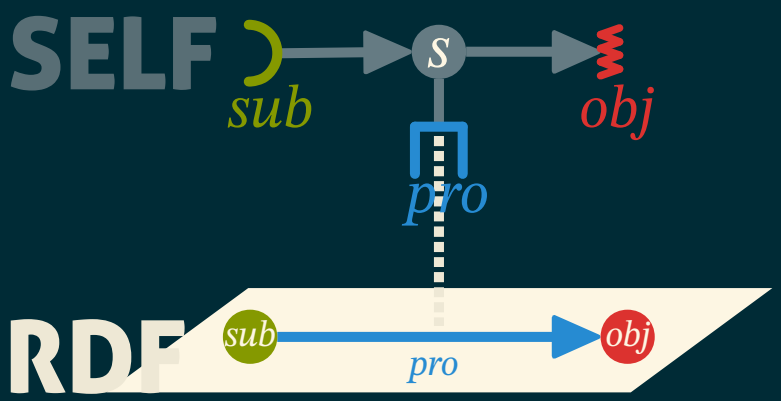
```
<?xml version="1.0"?>

<RDF>
  <Description about="Bob">
    <likes>Tea↙likes>
    <location>Kitchen↙location>
  ↙Description>
↙RDF>
```

- Languages
  - RDF
  - OWL-(Lite, DL, Full)
  - …
- Issues
  - Higher order knowledge
  - Modal Logic
  - Flexibility of the structure

SELF

- Defined by structure
- More expressive
- Allows complex data
- Fit for modal logic
  - Used in planning for fluents and states
  - Used in HTN for methods



Examples:

```
f = (bob @ kitchen);

a pre f;

a methods
{go(kitchen) → take(cup)};
```

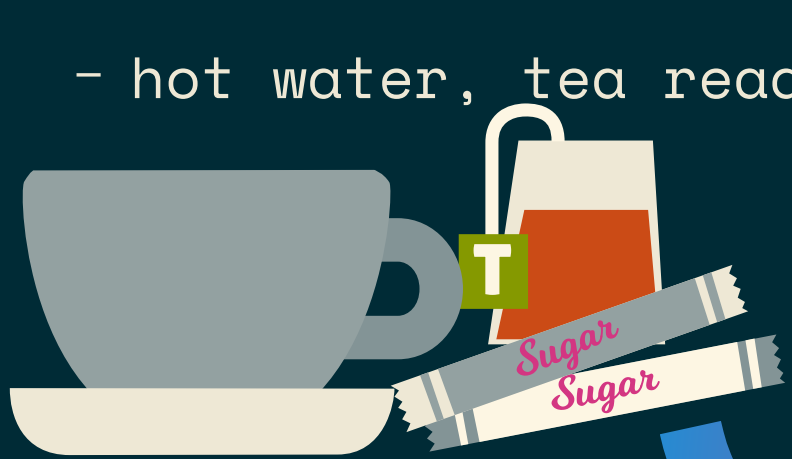
4 General Planning

Classical Planning

- Domain
  - Fluents
    - Formula over objects
  - States
    - Properties of the world
    - Formula over fluents
  - Actions
    - Precondition
    - Effects
- Problem
  - Initial state
  - Goal state
- Plan (solution)
  - Action sequence
  - Order
    - Total
    - Partial

Example

- Having some tea, aren't we ?
- Fluents
  - thing taken
  - hot water, tea ready
- Actions
  - take, brew, boil, …

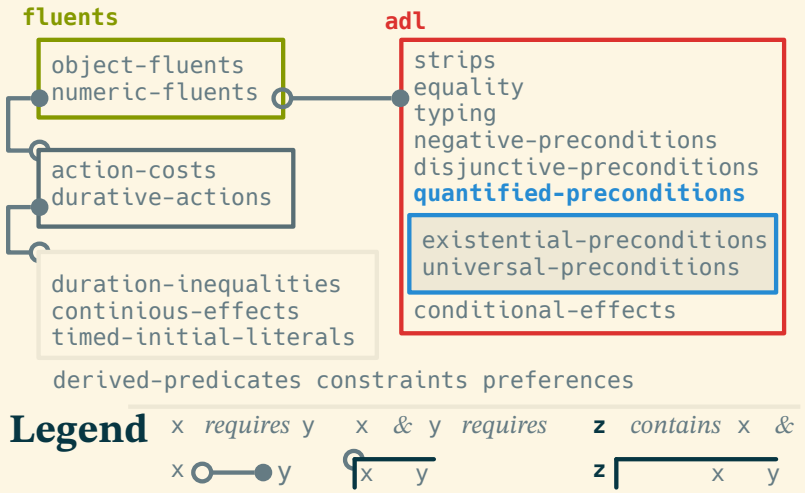


Initial State

Goal State

Existing Frameworks

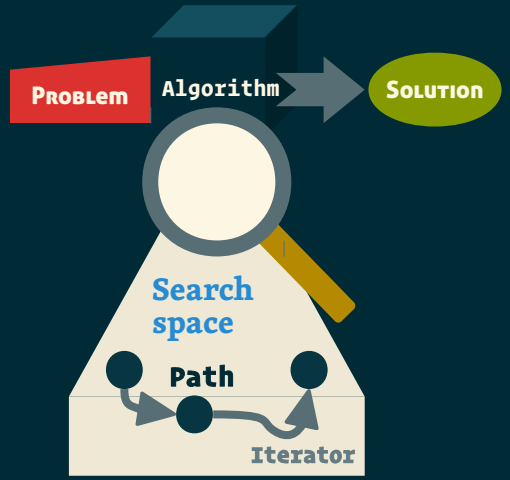
- PDDL:
  - Numerous extensions to the language
  - Not used in probabilistic or HTN planning
  - Most of the time translated into an intermediate language for planners



- Temporal
  - PDDL+
  - ANML
- Probabilistic
  - PPDDL
  - RDDL
- Multi-Agent
  - MAPL
  - MA-PDDL
- Hierarchical
  - UMCP
  - SHOP2
  - HDDL
  - HPDDL
- Ontological
  - WebPDDL
  - OPT
- Hybrids
  - SIADEX

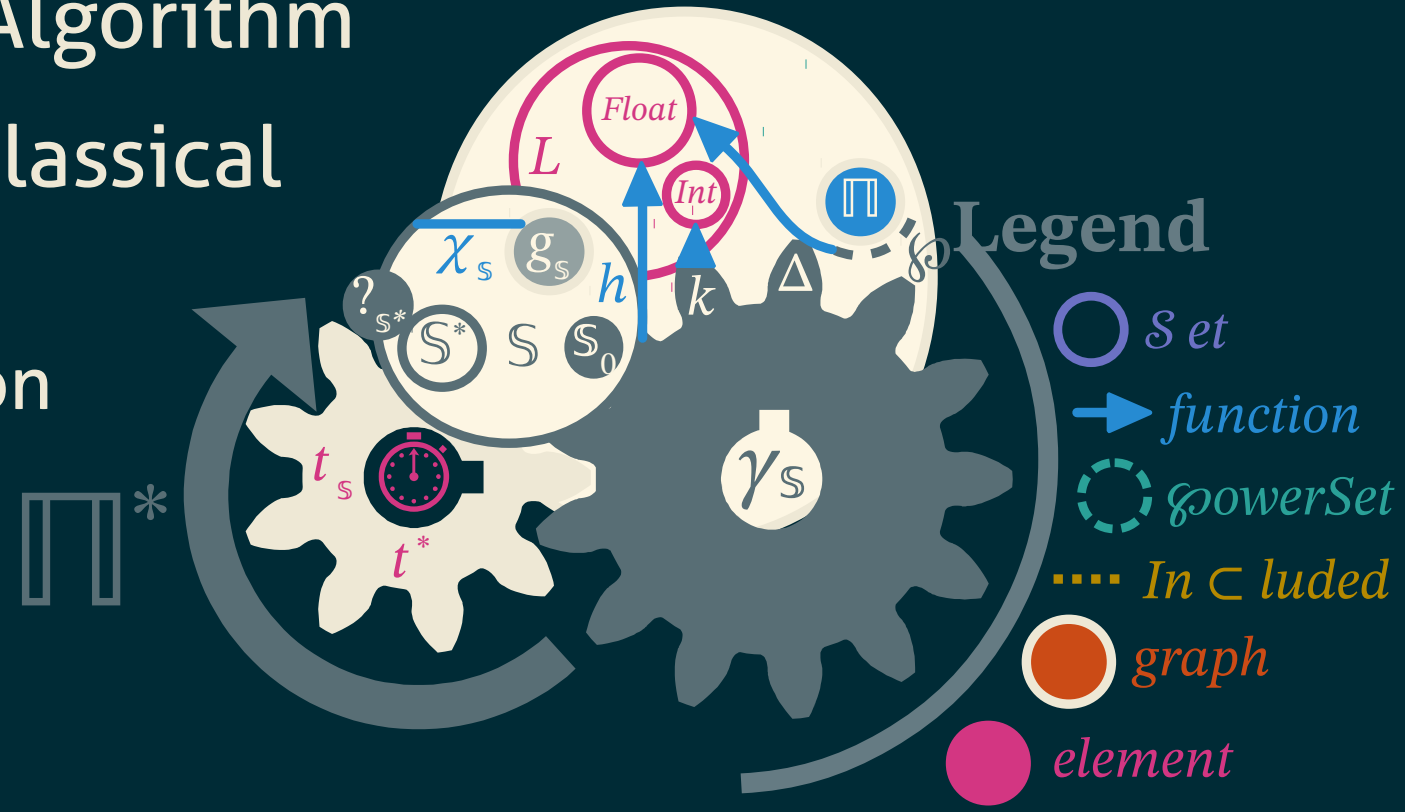
Planning Formalism Revisited

- States
  - And/Or trees of Fluents
  - Verifying
  - Applying
- Actions
  - Precondition, Effects
  - Constraints
  - Cost, Duration, Probability
  - Methods
- Search Space
  - Starting point
  - Iterator
  - Solution predicate



General Planning Algorithm

- Shortest Path Algorithm
- Instances for Classical Approaches
  - State-transition
  - Plan space
  - Case based
  - Probabilistic
  - Hierarchical



Π^*

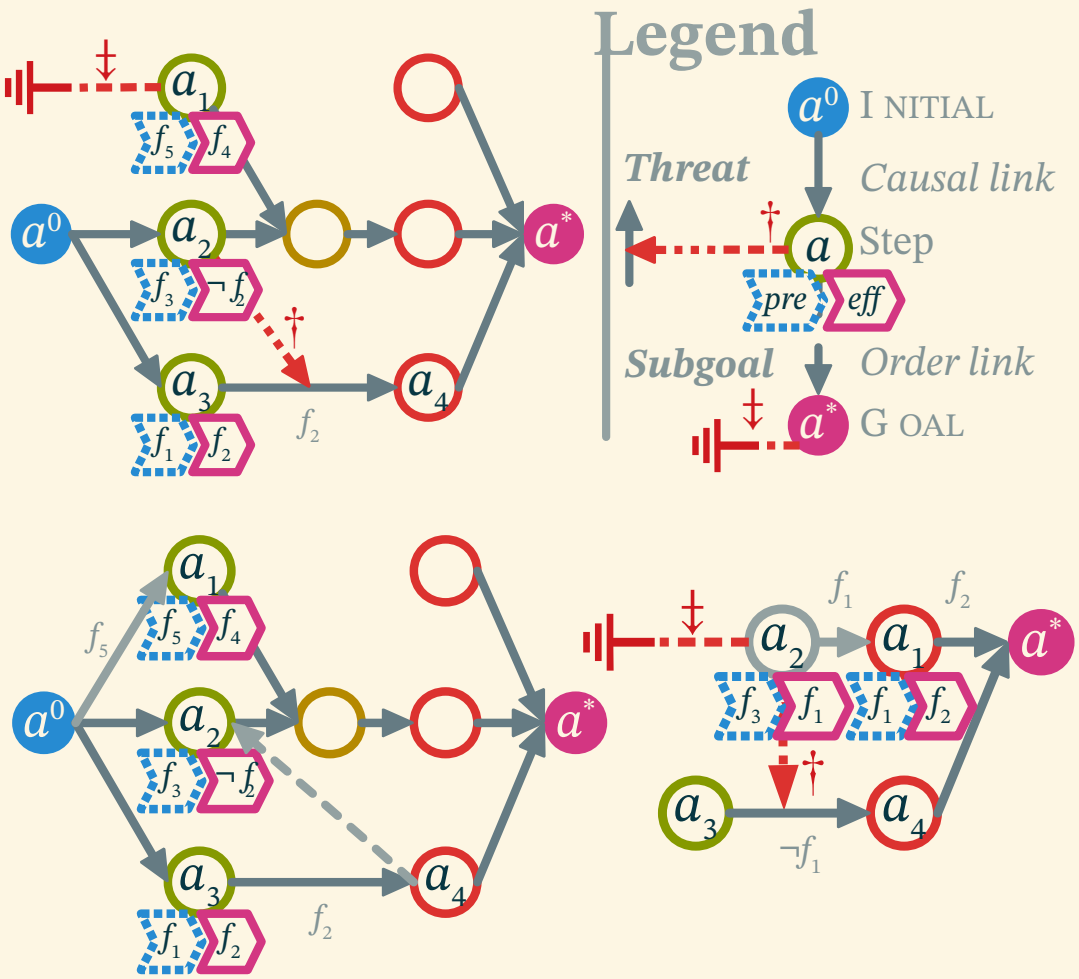
COLOR Framework

5 Flexible Online Planning

Planning Phases

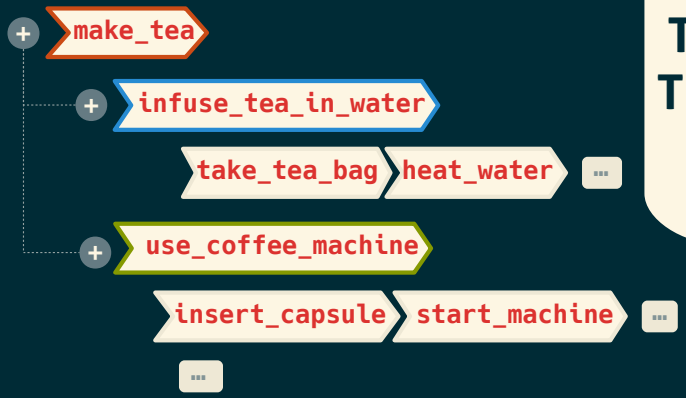
- Phases dependent on
  - Available information
  - Timing constraints
  - Planning paradigm

Plan Space Planning

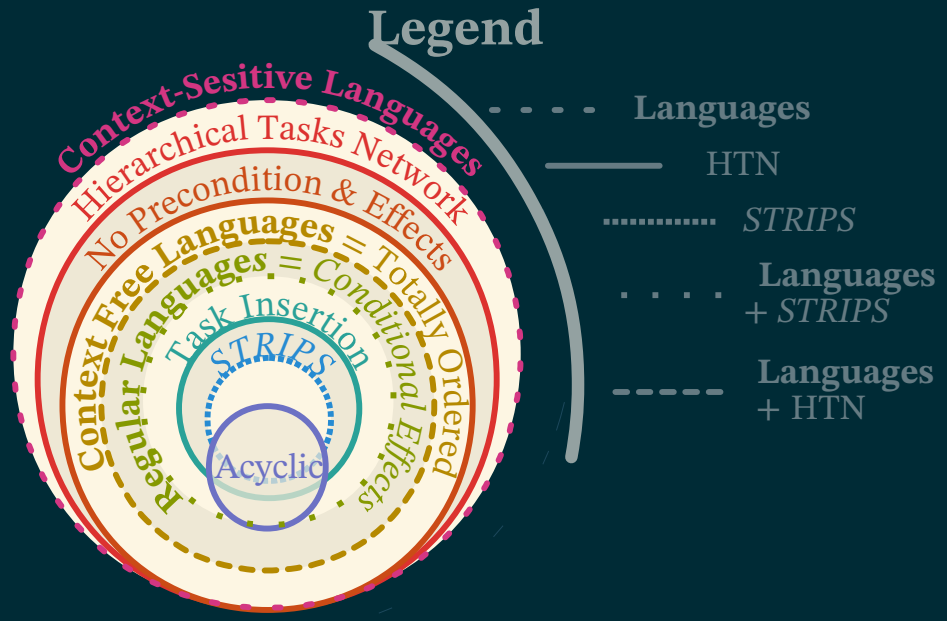


Hierarchical Task Networks

- Based on tasks
- Decomposition
- Vary in complexity

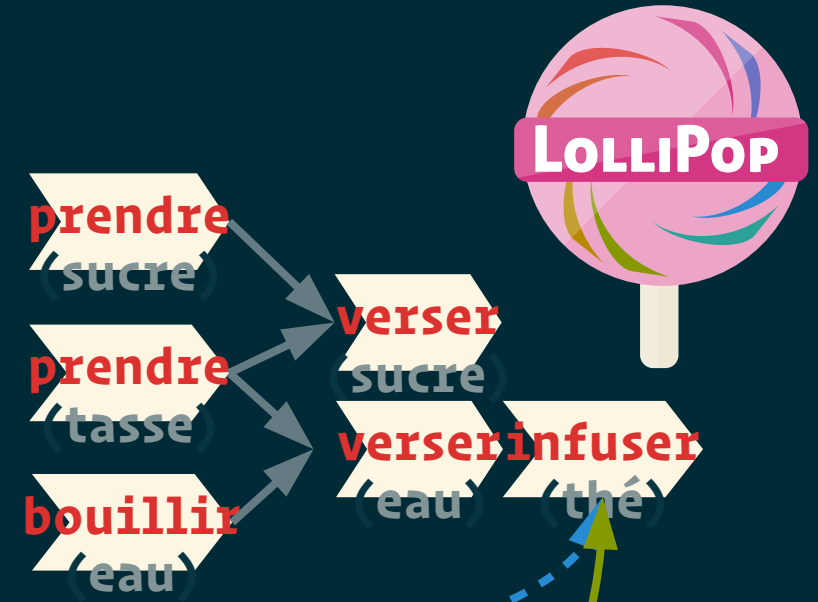


TODO : Citation of The difference with planning



Plan Repair Prototype

- Partial Order Planner (POP)
- Operator dependency graph
- Negative refinements
- Alternatives & Orphans
- Utility Heuristics

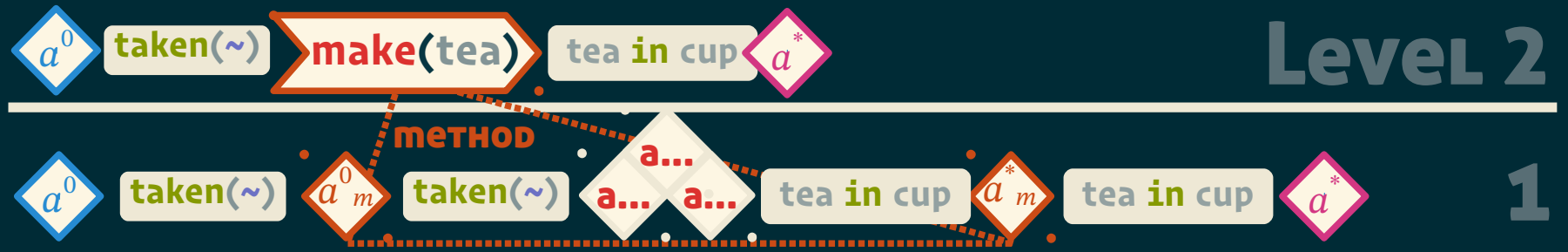


FIXME: Font size+ English

Abstract Planning

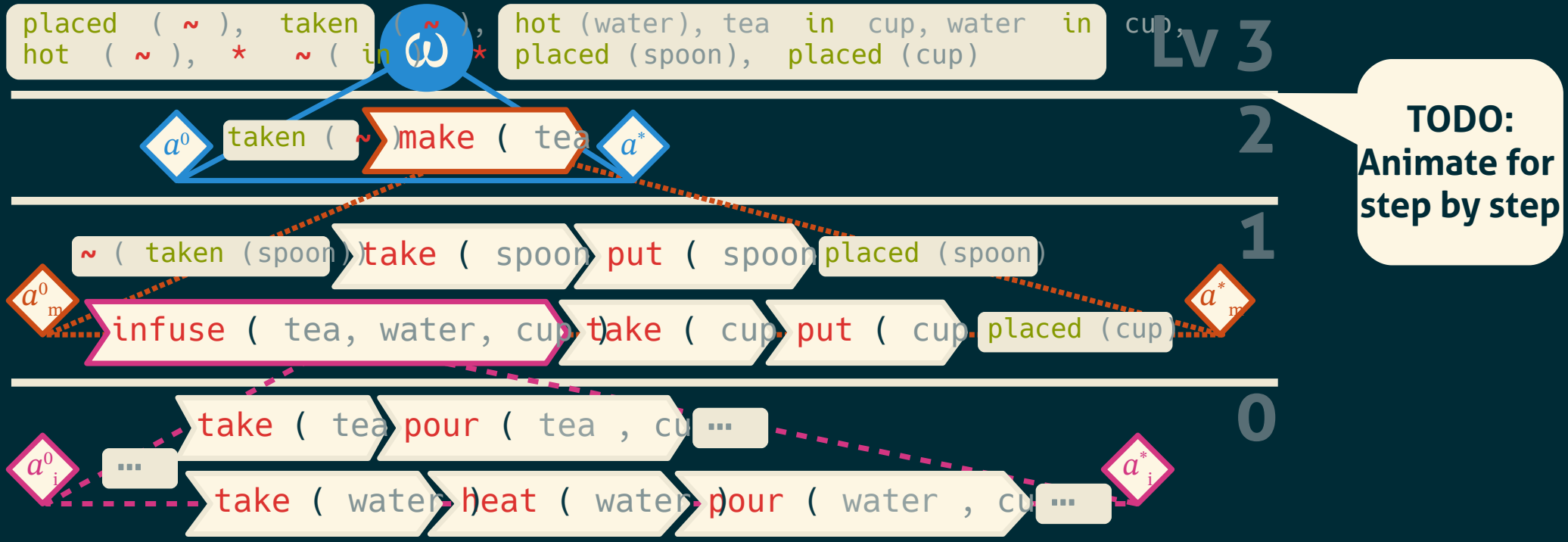
- HTN + POP planning
- Partial Resolution
  - An abstract solution at every level of abstraction
- Search by level
  - Expansion after completion :

- Decomposition flaw
  - Resolver : Decompose one composite action in the plan



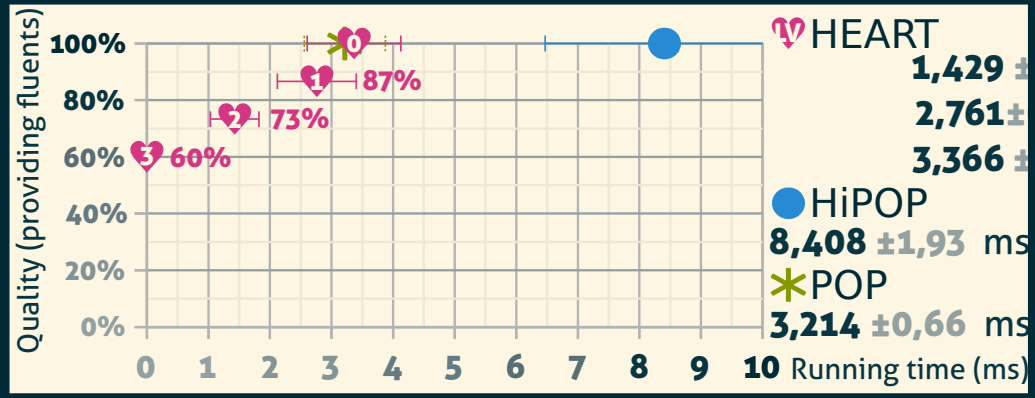
HEART

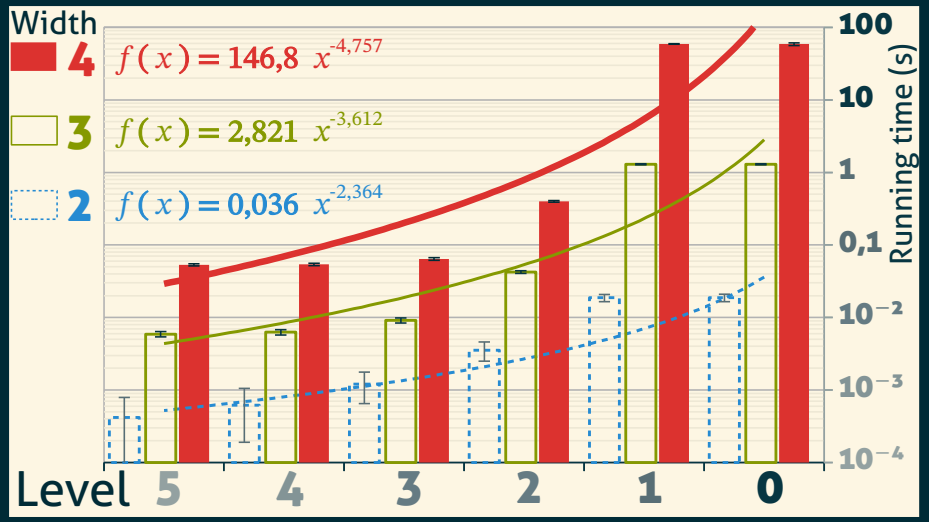
- Low priority for expansion
- Each level is a plan (abstract solution)
- Change of level
  - Propagation of atomic actions
  - Expansion of Composite Equities



Results

- 60% of the fluents before planning
- Exponentially faster at high abstraction levels
- Faster than HiPOP on some problems
- Common problems solved in milliseconds!





Toward Intent Recognition

- Formalize the linearization process
- Implement forward chaining version
- Test on more applied cases

6 Conclusion

Contributions & Results

- SELF: A knowledge description language defined by structure
- COLOR: A general framework for planning with its formalization
- LOLLIPOP: A plan repair planner for online planning
- HEART: A flexible approach to real-time planning for abstract planning

Perspectives

- SELF Improvement
  - Simplify the instantiation workflow
  - Allow for different amount of flexibility/performances
  - Test and improve performance and queries
- Planning Colorized
  - Conversion tool from PDDL
  - Make a clean implementation for community use
- Fixing Planning Domains
  - Allow HEART to discover new HTN methods (macro-action learning)
  - Make a debug tool to improve domains by logging dead-ends
  - Benchmark HEART and explore heuristics

Thanks for listening !