Robotics II - Planning

Lecture 2/3

M.Sc. Oscar Lima

November, 21th 2019

Task Planning Representations

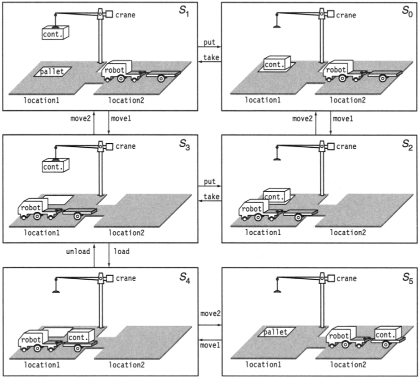
- Solution to the planning problem is performed via search
- How does the graph look like?

	Nodes	Edges
State Space	World states	State transition
Plan Space	Partial plans	Plan refinement operation
Task Networks	Primitive operator or Methods	Ordering constraints

- NOTE: HTN requires (additionally) a state space representation

State Space search

- Idea: Apply standard search algorithms (e.g. BFS, DFS, A*) to planning problem
- Nodes correspond to world states
- Arcs correspond to state transitions
- Path in the search space corresponds to plan



Forward Search pseudocode[1]

```
Forward-search(O, s0, g)
    s ← s0
    π ← the empty plan
    loop
        if s satisfies g then return π
        applicable ← {a | a is a ground instance of an operator in O,
                              and precond(a) is true in s}
        if applicable = ∅ then return failure
        nondeterministically choose an action a ∈ applicable
        s ← γ(s, a)
        π ← π.a
```

[1] Chapter 4, p 70 Dana Nau et al, Automated Planning: Theory & practice book

General idea:

- Start (backwards) from the goal state, $S \leftarrow g$
- Apply inverse of the planning operator (regression) to produce subgoals: $\gamma^{-}1(g, a)$
- Termination condition: $s_0 \subset S$, (goal is s_0)

Relevance:

- An action $a \in A$ is **relevant** iff:
  - $g \cap effects(a) \neq \{\}$
  - $g^{+} \cap effects^{-}(a) = \{\}$
  - $g^{-} \cap effects^{+}(a) = \{\}$

Regression:

- $\gamma^{-}1(g, a) = (g - effects(a) \cup precond(a))$

```
Backward-search(O, s_0, g)
    π ← the empty plan
    loop
        if s_0 satisfies g then return π
        relevant ← {a | a is a ground instance of an operator in O
                            that is relevant for g}
        if relevant = ∅ then return failure
        nondeterministically choose an action a ∈ applicable
        π ← a.π
        g ← γ^{-1}(g, a)
```

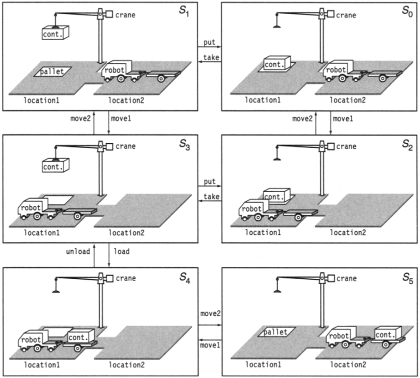
[2] Chapter 4, p 73 Dana Nau et al, Automated Planning: Theory & practice book

Assume:

- $g = S_4$, $s_0 = s_0$

Provide:

- ~~The set of applicable actions in s_0 (Fwd search)~~ (Too easy!)
- The set of relevant actions in g (Backward search)
- Output of the regression function for g (choose one action randomly from previous step)



- Nodes in graph represent partial plans
- Edges represent plan refinement operations
- $\pi \leftarrow \emptyset$, then we add 2 dummy actions (start, end)
- Start action has no preconditions and has s_0 as effects
- End action has goal state as precondition and no effects
- Goal: remove flaws from the plan while preventing threats

PDDL - Planning Domain Definition Language

Motivation and Introduction

- Do not write C++ code, just describe your domain!
- Standardize planners input → benchmark
- Created in 1998 by Drew McDermott and colleagues
- Inspired by STRIPS and ADL (Action Description Language)
- Used in International Planning Competition (IPC) 1998 / 2000

```
(define (domain cleaning_robot)

(:requirements
        :typing
)

(:types
        location
        robot
)

(:predicates
        (at ?r - robot ?l - location)     ; robot r? is at location l?
        (clean ?l - location)             ; location ?l is clean
)
```

[3] Source code: https://github.com/oscar-lima/pddl_problems/tree/master/cleaning_robot

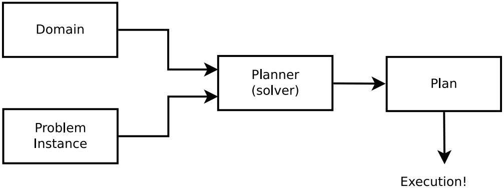
```
(:action move
        :parameters (?r - robot ?source ?destination - location)
        :precondition (at ?r ?source)
        :effect (and   ( not (at ?r ?source))
        (at ?r ?destination))
)

(:action clean
        :parameters (?r - robot ?l - location)
        :precondition (and (at ?r ?l) (not(clean ?l)))
        :effect (clean ?l))
)
```

Syntax - Problem


```
(define (problem p01)

(:domain cleaning_robot)

(:objects
        ghost - robot
        locA locB - location
)

(:init
        (at ghost locA)
        (not(clean locA))
        (not(clean locB))
)

(:goal
        (   and (at ghost locA)
                (clean locB)
                (clean locA)
        )
)
)
```

Elements

- domain.pddl
- problem.pddl
- try it! :
- http://editor.planning.domains



- Exercise: Model the transfit scenario
- Storyline: The robot has to bring a solar panel autonomously from the packing box
- Define your types
- Draw your scenario
- Bring actions
- make preconditions and effects
- e.g.

Tools

- https://github.com/nergmada/planning-wiki

Examples

- Multiple examples available under:
- https://github.com/oscar-lima/pddl_problems
- https://github.com/primaryobjects/strips

- PDDL 1.2 : Classical planning
- PDDL 2.1 : Temporal planning, numeric fluents, plan metrics, durative/continuous actions
- PDDL + : Allows modeling of mixed discrete-continuous domains (hybrid), allows processes, events, timed initial literals (TIL)

(some...) Task planning formalisms

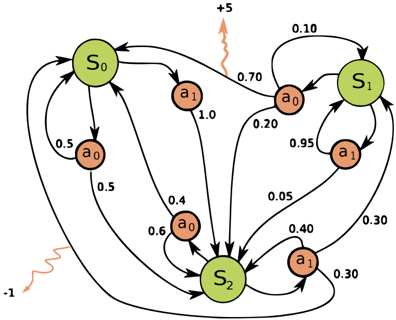
STRIPS[4]

- STRIPS is both a formalism and a planner

[4] Stanford Research Institute Problem Solver

PPDDL[7]

- Official language of IPC probabilistic track 2004 and 2006
- Extends PDDL 2.1 with probabilistic effects, rewards and goal rewards
- MDP[5] planning that allows uncertainty between state transitions
- Environment remained fully observable (no POMDP[6])



5 Markov Decision Process

6 Partially Observable Markov Decision Process

7 Probabilistic Planning Domain Definition Language

- A language to describe hierarchical planning problems [Höller et al. 2019]
- Recently proposed by researchers from Ulm University (2019)
- Will be used for the first time in IPC[8] 2020 [Behnke et al. 2019]
- Will allow to benchmark HTN planners
- Previously the HTN planning community lacked a common input language
- Built as an extension to PDDL
- e.g.

```
(:method m-drive-to-via
        :parameters (?li ?ld - location)
        :task (get-to ?ld)
        :subtasks (and
                (t1 (get-to ?li))
                (t2 (drive ?li ?ld)))
        :ordering (and
                (t1 < t2))
)
```

[8] International Planning Competition
[9] Hierarchical Domain Definition Language

RDDL[12]

- Used in last IPC probabilistic track (2018)
- Unlike PPDDL, RDDL can model partial observability (POMDP)
- Significantly different to PPDDL (syntactically and semantically)
- A grounded RDDL corresponds to a DBN[10]
- RDDL domain example code available here[11]

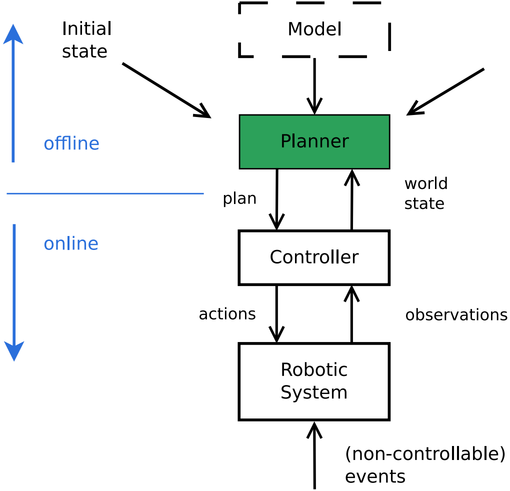
[10] Dynamic Bayesian Network

[11] https://github.com/KCL-Planning/rosplan_demos/blob/master/rosplan_demos/common/turtlebot_domain.rddl

[12] Relational Dynamic influence Diagram Language

Plan execution

Classic execution architecture



Scheduling

- Dispatch actions
- Encode plan as an STN
- e.g. Esterel dispatch (ROSPlan)

Environment representation

- Sense environment
- Transform into symbols
- Maintain KB (useful for re-planning)

ROSPlan

- Execution framework for Robotics
- Based on Robot Operating System (ROS)
- Maintained by KCL University
- Open source:
- https://github.com/kcl-planning/ROSPlan

Trends in Task planning

- Temporal planning
- Plan with deadlines
- TIL (Timed Initial Literals)
- Hybrid planning

- Critical for interaction with humans to enable cooperation
- Motivation: Trust, interaction, transparency [**fox2017explainable**]
- ICAPS Workshop in 2018[13]
- Relevant questions:
  - "How can the PS system explain particular steps, ordering decisions, or resource choices?" (XAIP)
  - "How can a PS system explain that no solution is possible, or what relaxations of the constraints would allow a solution?" (XAIP)
  - "why is taking so long to plan?"
- Potentially hypothetical reasoning could be used...

[13] http://icaps18.icaps-conference.org/xaip/

Others

- Risk-aware planning (bounded) [**santana2016risk**]
- Collaborate, HRI revise goals
- Long term autonomy
- Combine motion and task planning

Task Planning Venues

ICAPS[14]

- Largest Task Planning scientific **conference**
- Forum for researchers and practitioners in Planning and Scheduling
- Resulted from the merging of:
  - International Conference on Artificial Intelligence Planning and Scheduling (AIPS)
  - The European Conference on Planning (ECP)
- Robotics track available!
- Next ICAPS 2020: Nancy, France
- Website: http://www.icaps-conference.org/

[14] International Conference on Automated Planning and Scheduling

IPC[15]

- Organized as part of ICAPS conference
- Empirically evaluates state-of-the-art planning systems on a number of benchmark problems
- All planners that participate need to be open source after competition
- Last IPC was in 2018 and considered:
  - Deterministic track
  - Probabilistic track
  - Temporal track
- Website: http://www.icaps-conference.org/index.php/Main/Competitions

[15] International Planning Competition

PlanRob Workshop

- Organized as part of ICAPS conference
- Intersection between Task Planning & Scheduling and Robotics
- Usually co-organized by Dr. Michael Cashmore (ROSPlan creator)
- Concerned among other topics with real-world planning applications for autonomous and intelligent robots
- Last PlanRob website: https://icaps19.icaps-conference.org/workshops/PlanRob

Some planning research groups

This slide contains clickable links:

- MIT CSAIL MERS (Prof. Brian Williams)
- KCL (Andrew & Amanda Coles, previously Derek Long and Maria Fox)
- DFKI Niedersachsen PBR - Osnabrück (Prof. Dr. Joachim Hertzberg)
- Orebro AASS (Prof. Alessandro Saffiotti, Prof. Federico Pecora)
- Saarland FAI (Joerg Hoffmann)
- LAAS CNRS RIS (Malik Ghallab)
- Uni Freiburg (Prof. Dr. Bernhard Nebel)

Next lecture contents

- Motion Planning
  - Path planning (navigation 2D)
  - Motion planning (robot manipulators)
  - Grasp planning (robot end effector)

Thank You for Your Attention.
Do You Have Questions?

References I

[Höl+19] Daniel Höller et al. "HDDL-A Language to Describe Hierarchical Planning Problems". In: (2019).

[Beh+19] G Behnke et al. "Hierarchical Planning in the IPC". In: *Proc. of the Workshop on the IPC (WIPC)*. 2019.