6 - Width Based Planning, Plan & Goal Recognition

知识点 & 题目

Width Based Planning

A new width notion and planning algorithm exponential in problem width:

- Benchmark domains have small width when goals restricted to single atoms
- Joint goals easy to serialize into a sequence of single goals

Do you want Hard Problems?

- problems with high atomic width (apparently no benchmark in this class)
- multiple goal problems that are not easy to serialize (e.g. Sokoban)

Key definition: the **novelty $w(s)$ of a state** s is the size of the smallest subset of atoms in s that is true for the first time in the search.

- e.g. $w(s) = 1$ if there is **one** atom $p \in s$ such that s is the first state that makes p true.
- Otherwise, $w(s) = 2$ if there are **two** different atoms $p, q \in s$ such that s is the first state that makes $p \wedge q$ true.
- Otherwise, $w(s) = 3$ if there are **three** different atoms...

- **atoms -> facts F**
- Novelty table

Iterated Width (IW)

Algorithm

- $IW(k)$ = breadth-first search that prunes newly generated states whose $novelty(s) > k$.
- IW is a sequence of calls $IW(k)$ for $i = 0, 1, 2, \ldots$ over problem P until problem solved or i exceeds number of variables in problem

Properties

$IW(k)$ expands at most $O(n^k)$ states, where n is the number of atoms.

- While simple and blind, performs well over benchmarks when goals restricted to single atoms
- This is no accident, width of benchmarks domains is small for such goals

Key theory of IW(k) in terms of width:

Properties

For problems $\Pi \in \mathcal{P}$, where $width(\Pi) = k$:

- $IW(k)$ solves Π in time $O(n^k)$;
- $IW(k)$ solves Π optimally for problems with uniform cost functions
- $IW(k)$ is complete for Π

Theorem

Blocks, Logistics, Gripper, and n-puzzle have a bounded width independent of problem size and initial situation, provided that goals are single atoms.

In practice, $IW(k \leq 2)$ solves 88.3% IPC problems with single goals:

Serialized Iterated Width (SIW)

- Simple way to use IW for solving real benchmarks P with joint goals is by simple form of "hill climbing" over goal set G with |G| = n, achieving atomic goals one at a time
- SIW uses IW for both decomposing a problem into subproblems and for solving subproblems
- It's a blind search procedure, no heuristic of any sort, IW does not even know next goal G_i "to achieve"

IW: sequence of novelty-based pruned breadth-first searches

- Experiments: excellent when goals restricted to atomic goals
- Theory: such problems have low width w and IW runs in time $O(n^w)$

SIW: IW serialized, used to attain top goals one by one

- Experiments: faster, better coverage and much better plans than GBFS planner with h_{add}
- Intuition: goals easy to serialize and have atomic low width w

Balancing exploration and exploitation

State-of-the-art methods for satisficing planning rely on:

- heuristics derived from problem
- plugged into Greedy Best-First Search (GBFS)
- extensions (like helpful actions and landmarks)

GBFS is pure greedy "exploitation"; often gets stuck in local minima

- Recent approaches improve performance by adding exploration

Exploration required for optimal behavior in RL and MCTS

- Such methods perform flat exploration that ignores structure of states

Best-First Width Search (BFWS)

BFWS(f)

BFWS(f) for $f = \langle w, f_1, ..f_n \rangle$ where w is a novelty-measure, is a plain best-first search where nodes are ordered in terms of novelty function w, with ties broken by functions f_i in that order.

Basic BFWS($\langle w, h \rangle$) scheme obtained with $\mathbf{h} = \mathbf{h_{add}}$ or $\mathbf{h_{ff}}$, and novelty-measure $\mathbf{w} = \mathbf{w_h}$, where

- $w_h(s)$ = size of smallest new tuple of atoms generated by s for the first time in the search relative to previously generated states s' with $h(s) = h(s')$.

$\rightarrow$ BFWS($\langle w, h \rangle$) much better than purely greedy BFS(h)

Models and Simulators

Simulators: without a representation of action preconditions and effects

- Developing a planner that uses action structure only to define
  - the set A(s) of applicable actions in state s
  - state transition function f(a,s)
- The planner does not see action preconditions and effects but just the functions A(s) and f(a,s)
- Its performance matches the performance of state of the art planners that make use of PDDL representations, over the existing PDDL benchmarks

Modelling

Many problems fitting classical planning model but difficult to describe in PDDL are easily modeled now: Pacman, Tetris, Pong, etc.

- Expressive language features easily supported: functions, conditional effects, derived predicates, state constraints, quantification, ...
- Any element of the problem can be modeled through logical symbols attached to external procedures (e.g. C++).
- Action effects can be given as fully-black-box procedure taking the state as input.
- Many problems fit Classical Planning model, but hard to express in declarative languages.

Simulated BFWS   L6 P25

- No need for planning languages that reveal structure of actions (e.g. action preconditions and effects)
- Not much efficiency appears to be lost in second pathway

Challenges

- Non-linear dynamics
- Perturbation in flight controls
- Partial observability
- Uncertainty about opponent strategy

Classical Planning with Simulators   L6 P29

Plan & Goal Recognition

Plan Recognition (PR) is Planning in reverse

- Planning - we seek plans to achieve goals G.
- PR: find goals G accounting for partially observed plan.

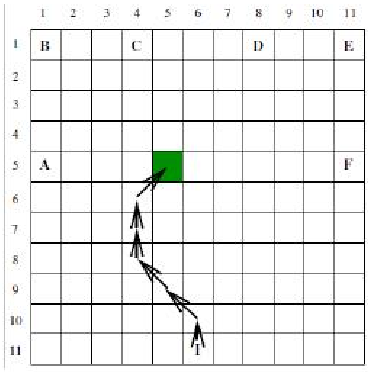
Formalising GR as a Multi-Agent Task

- Two possible roles for each agent:
    - Actor - performs actions to change the state of the world
    - Observer - perceives actions and updates its belief on the Actor intentions
- Three possible stances for the Actor
    - Adversarial - obfuscates deliberately its goals
    - Cooperative - tries to tell the Observer what she is up to
    - Indifferent - does not care about the Observer
- Open Challenge: Stances could be changing over time

Components of Goal Recognition Task

- Actions describe what the Actor does
    - Walking from X to Y , opening a door, using a credit card...
- Goals describe what the Actor wants
    - To have breakfast, Park a car, Wreck a web service...
- Plans describe how goals can be achieved
    - Ordered sequences of actions
    - These can be ranked according to cost or efficiency
- Sensor Model describes what does the Observer perceives
    - Does it always see every action done by the Actor?
    - Are actions observed directly? Or only their effects are?
    - Does it know exactly where in the world the Actor is?
- Goal Recognition can be modeled using STRIPS

Example: Agent on a Grid World   L6B P14



Counterfactual Reasoning (Pearl, 2001) to Establish Necessity

Compare **cost** of best plans that do not comply with observed actions, with best plans that do.

$\rightarrow$ Then it follows B and C *more likely* than A or the rest.

Key facts of the Model-Based Approach

1. Π given **implicitly**, requires to **solve** $|\mathcal{G}|$ planning tasks
2. Plans "**extracted**" with **off–the–shelf** planning algorithms.
3. **Plausibility** of goals $\mathcal{G}$ given as a **probability distribution**
   - Goals are *plausible* when motivate plans *consistent* with O,
   - and when O is *necessary* to achieve goals *efficiently*.

Roadmap

- Make off-the-shelf (现成的) planners compute constrained w.r.t (with reference to) O
- Derive P(G|O) from best plans that comply with and work around O

PR as planning: Inferring the Goal Probabilities

Goal

Obtain **probability distribution** $P(G|O)$, $G \in \mathcal{G}$.

Outline of Approach

From **Bayes' Rule** $P(G|O) = \alpha\, P(O|G)\, Prob(G)$, where

- α norm. constant
- $Prob(G)$ given in problem specification
- $P(O|G)$ function of extra cost needed to not comply with O

$$P(O|G) = \text{function}(c^*(P'[G+\overline{O}])) - c^*(P'[G+O]))$$

- cost to not complying with observations - cost to comply with observations

Goals as Predictors for O (informally)

- G predicts O badly when it would be more efficient to deviate from O.
- G predicts O perfectly when G unfeasible if not doing O.

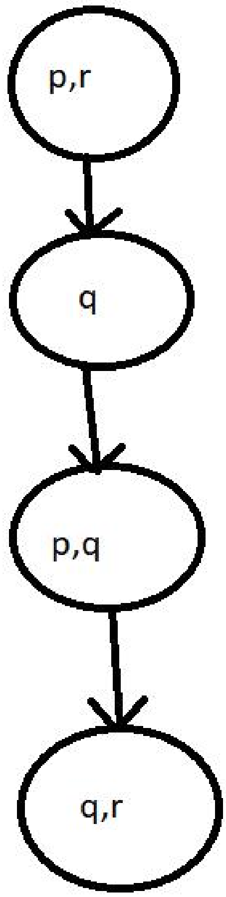
题目

Quiz

Question 1

1 / 1 pts

What is the novelty value of the last state if F = {p,q,r}?



- 1
- Correct! 2
- 3
- |F| + 1

Question 2

1 / 1 pts

IW(1) is a Breadth First Search that prunes generated nodes with novelty > 1. IW(1) can solve the TSP problem

- True
- Correct! False