CompSci 367, tutorial 8

Planning assignment

Alex Donaldson

Planning overview

- Planning is a specific type of state space search which makes use of additional information about the task
- State represented by literals (true xor false)
- Problem is composed of:
  - Literals (propositional expressions – use predicates)
  - Initial state (fully defined)
  - Goal state (can be partially defined)
  - Operators (aka actions), which have preconditions and effects

Planning overview

- 3 different languages
  - State language – for representing a state
  - Goal language – for representing goals, and preconditions of operators, and derived predicates
  - Update language – for representing effects of operators
- + Prolog for meta-level predicates

Planning overview

- State representation uses:
  - state language
- Goal representation uses:
  - goal language
- Operator representation, has [name, set of parameters, preconditions, effects], uses:
  - goal language, for preconditions
  - update language, for effects

Planning overview

- State language, types of state predicates:
  - Polarity
    - negative (default)
    - positive
  - Changeability
    - fluent (default) = can change
    - static = can’t change for a given problem

Planning overview

- Goal language adds derived and meta-level predicates:
  - Value access
    - primitive (default) = values can’t be found from other predicates, must be in state description
    - derived = values found from other predicates, not stored in state description
  - Value basis
    - object level (default) = tests a state
    - Meta-level = state is not directly used (Prolog, not state language). Motivation: Object level predicates can only access literals, so if we want to do some other operations e.g. < comparison, we have to use a different type of predicate and language - meta-level predicates and Prolog.

Planning overview

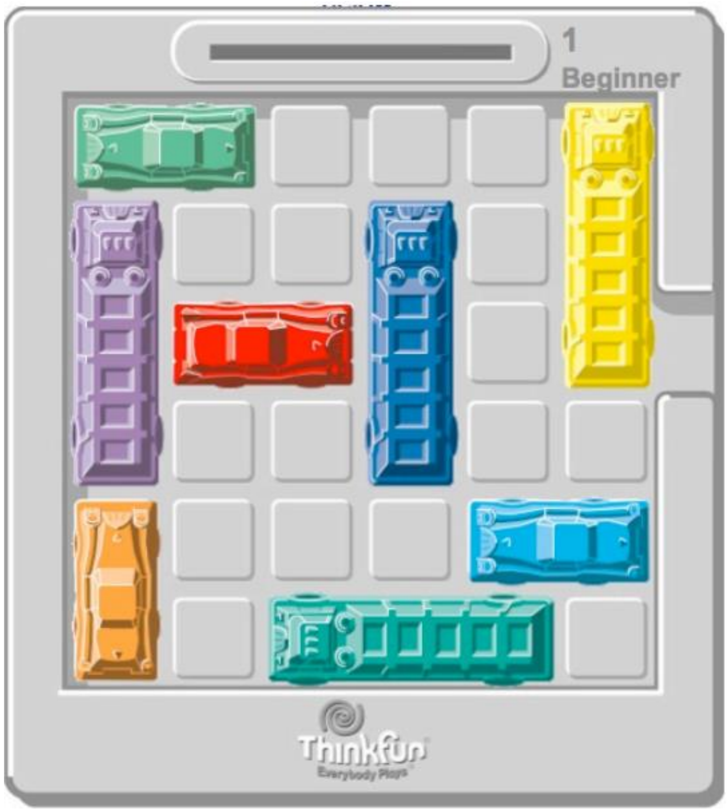
- Goal language
  - *don’t care* is default, positive and negative both need to be specified
- Update language
  - *don’t change* is default
  - positive literals added to state description
  - negative literals removed from state description

Assignment overview

- Provided files do domain independent planning (i.e. you specify the domain)
- Example files for blocks world domain are provided
- For RushHour, primitive predicates are provided
- You need to specify other predicates, and operators

Assignment overview

- RushHour



Assignment overview

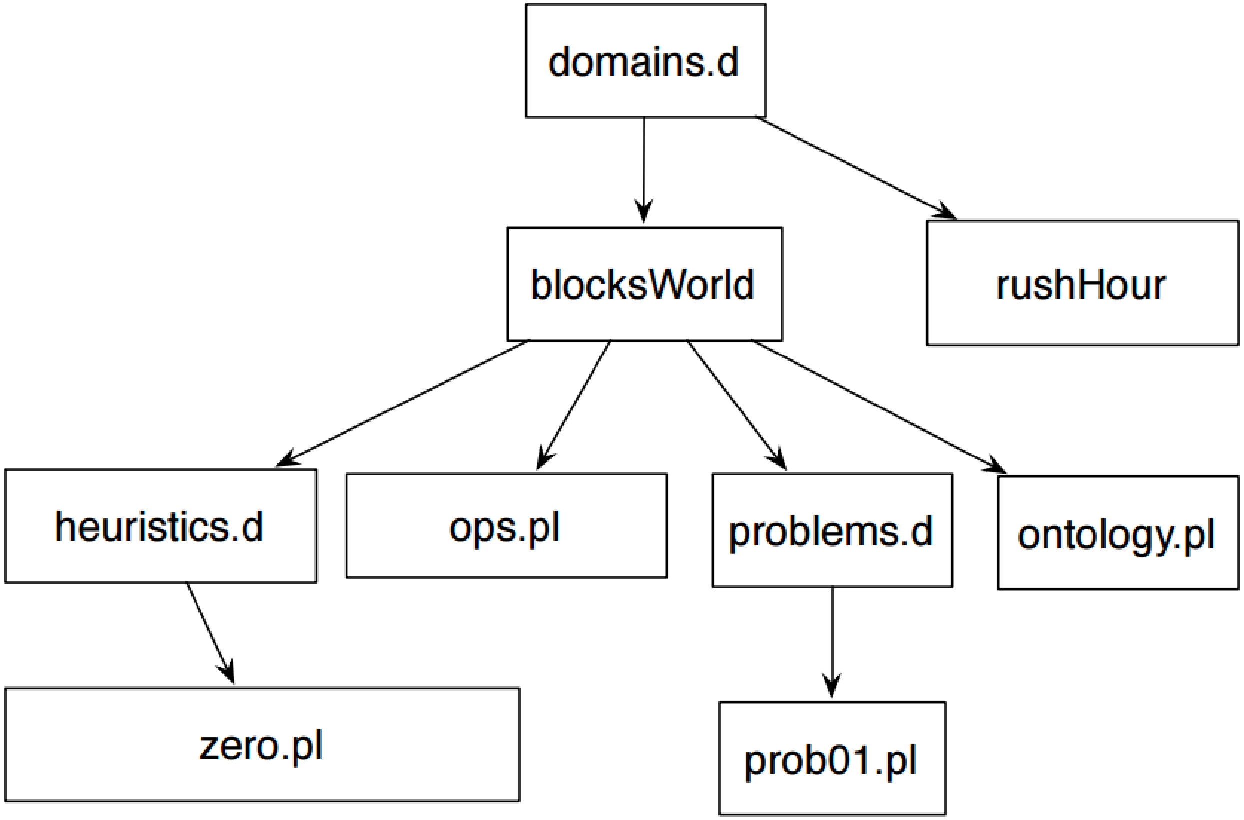
- Primitive predicates:
  - *hoodAt/2* e.g. *hoodAt(blueTruck, 9)*
  - *trunkAt/2* e.g. *hoodAt(blueTruck, 21)*
  - *width/1*, width of the grid
  - *height/1*, height of the grid

Assignment overview

- Only 2 actions:
  - *move,* move a vehicle one grid square in the direction it is facing
  - *reverse,* swap locations of the front and back of the vehicle

Assignment overview

- Directory structure



domains.d

blocksWorld

rushHour

heuristics.d

ops.pl

problems.d

ontology.pl

zero.pl

prob01.pl

Demonstration with blocks world