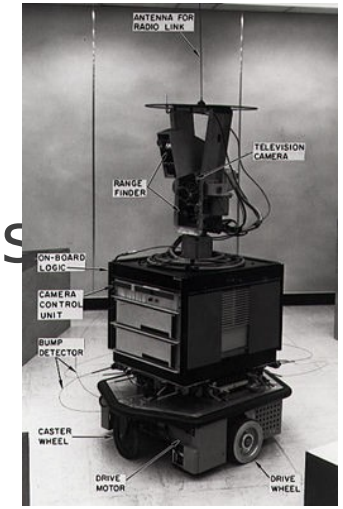
PDDL and Planners

PAH (Planning and Games)

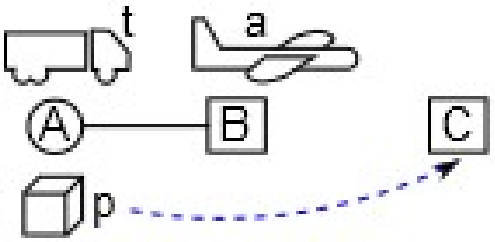
Michal Štolba
michal.stolba@agents.fel.cvut.cz

STRIPS (**St**anford **R**esearch **I**nstitute **P**roblem **S**olver)



- 1966-1972 – Shakey the Robot
- <P,O,I,G>
  - P – finite set of propositional (true/fals variables
  - O – finite set of operators:
  - pre: {p∈P s.t. p=true}
  - add: {p∈P s.t. p←true}
  - del: {p∈P s.t. p← false}
  - I – initial state (p∈P s.t. p=true, other false)
  - G – goal state (p∈P s.t. p=true; p∈P s.t. p=false)
- Set representation

STRIPS - Example



- *P* – propositions:
  - truck-at-A, truck-at-B
  - plane-at-B, plane-at-C
  - package-at-A, package-at-B, package-at-C, package-in-t, package-in-a
  - 2^9 = 512 states
- O – operators:
  - load-p-a-B
  - pre: {plane-at-B, package-at-B}
  - add: {package-in-a}; del: {package-at-B}

Multi-valued Planning Task (MPT or SAS+)

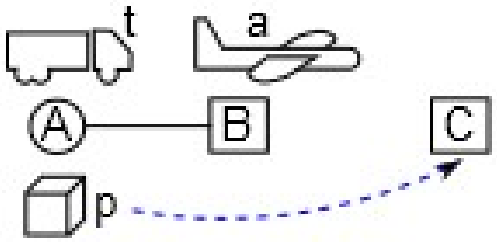
- 1995 (SAS+), 2005 (MPT – Fast Downward)
- $<V,i,g,O>$
  - V – finite set of state variables v with associated finite domain Dv
  - partial state over V is a function s over some subset of V s.t. $s(v) \in Dv$ whenever $s(v)$ is defined
  - state is a partial state s.t. s is defined for all $v \in V$
  - i – state over V called initial state
  - g – partial state over V called goal state

Multi-valued Planning Task (continued)

- $<V,i,g,O>$
  - O – finite set of operators <pre,eff>
  - pre: partial assignment (state) over V
  - eff: $<cond, v, d>$
  - *cond*: (possibly empty) partial assignment over V
  - $v \in V$ – affected variable
  - $d \in Dv$ – new value for v

- Plan existence PSPACE-complete
- Automatic conversion from STRIPS

MPT - Example



- *V* – variables and their domains:
  - truck-at ∈ {A,B}
  - plane-at ∈ {B,C}
  - package-at ∈ {A,B,C,t,a}
  - 2x2x5 = 20 states
- O – operators:
  - load-p-a-B
  - pre: plane-at=B, package-at=B
  - eff: <{}, package-at, a>

PDDL (Planning Domain Definition Language)

- General language to describe planning problems
  - Domain – definition of types, predicates, operators
  - Problem – definition of objects, initial state and goal
  - Lisp-like syntax
  - Prefix notation (+ 1 2)
  - A lot of brackets
  - Several versions (1.2, 2.1, 3.1)

Planners (1)

- FF (Fast Forward, 2001)
  - Forward-chaining heuristic state space search
  - Enforced hill-climbing / Breadth-first search
  - FF heuristic
- FD-fdss (stone soup)
  - Fast Downward (2006)
  - MPT, several search strategies, several heuristics
  - Automatic configuration

Planners (2)

- FD-ms
  - Fast Downward
  - A* + Merge&Shring abstraction heuristic
- FD-lmcut
  - Fast Downward
  - A* + LM-Cut heuristic

Planners (3)

- Lama 2011
  - Built on FD
  - Multi-heuristic search (FF, Landmarks)
  - Weighted A*
- PROBE
  - GBFS + h_add
  - From each state a greedy probes with highly informed heuristics

Planners (4)

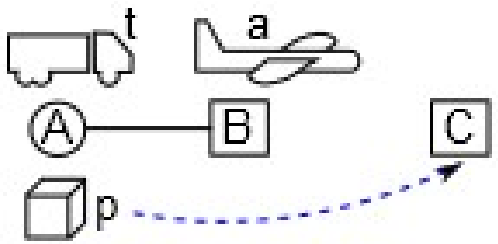
- SymBA*
  - Bidirectional A*
  - Perimeter-based abstraction heuristic
- Mercury
  - GBFS
  - Red-black relaxation heuristic

Planners (5)

- yahsp3
  - Heuristic search with lookahead using relaxed plans

PDDL Excercise

- Formalize:



- Run planners
- Extend