Introduction to classical planning

Problem representations

Michaela Urbanovská

PUI Tutorial
Week 1

Course organization

- Classical planning (M. Urbanovska), probabilistic planning (J. Mrkos)
- Each part one assignment - 50 points total
- Classical planning assignment → programming your own **planning system**
  - C, C++, Java
  - more details later
  - Deadline: half of April (TBD)
- **Zapocet:** 25 out of 50 points to pass
- **Exam:** 25 out of 50 points to pass
- **Final grade:** zapocet points + exam points
- Problems? Questions?
  - **urbanm30@fel.cvut.cz**

Tutorials

- STRIPS, FDR problem definitions
- PDDL, compilations
- Relaxation heuristics
- Landmark heuristics
- Abstraction heuristics
- LP based heuristics
- Machine learning in planning
- ...and much more!

Lecture check

- Any questions regarding the lecture?

Lecture check

- Any questions regarding the lecture?

Planning motivation

- **General problem solving**
- Basically can solve all your problems

Planning motivation

- **General problem solving**
- Basically can solve all your problems
- Problem

Planning motivation

- **General problem solving**
- Basically can solve all your problems
- Problem + representation

Planning motivation

- **General problem solving**
- Basically can solve all your problems
- Problem + representation + solver

Planning motivation

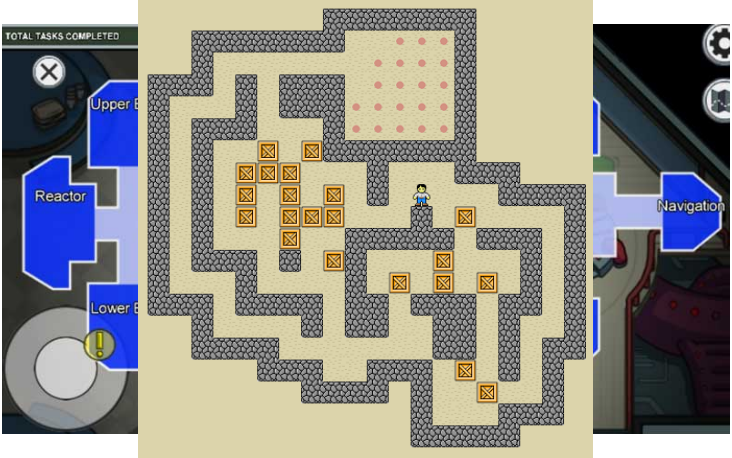
- **General problem solving**
- Basically can solve all your problems
- Problem + representation + solver = **solution**

Planning motivation

- **General problem solving**
- Basically can solve all your problems
- Problem + representation + solver = **solution**

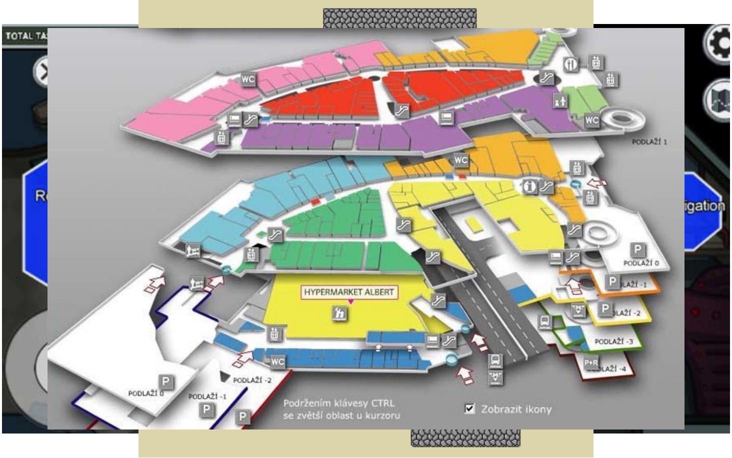
Planning motivation

- **General problem solving**
- Basically can solve all your problems
- Problem + representation + solver = **solution**



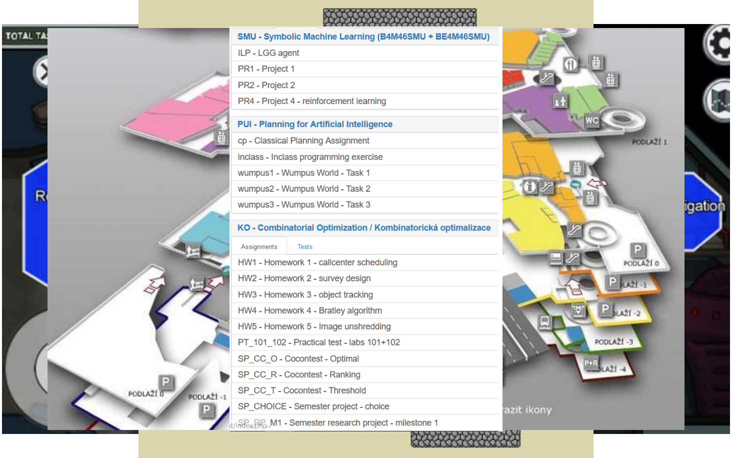
Planning motivation

- **General problem solving**
- Basically can solve all your problems
- Problem + representation + solver = **solution**



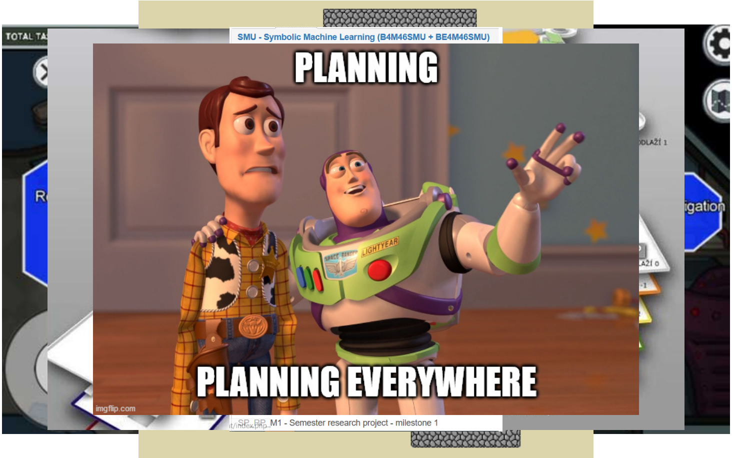
Planning motivation

- **General problem solving**
- Basically can solve all your problems
- Problem + representation + solver = **solution**



Planning motivation

- **General problem solving**
- Basically can solve all your problems
- Problem + representation + solver = **solution**

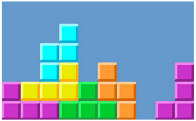


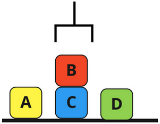
Planning benchmarks

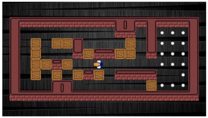
For example...

- Airport
- Depot
- Sokoban
- Blocksworld
- Elevators
- Floortile
- Parking
- Pipesworld
- Tetris
- Tidybot
- Woodworking









Benchmarks

- Often inspired by real-world problems
- Sometimes even modeled real-world problems
- Problems with interesting properties to test performance of algorithms heuristics
- Doesn't have to correlate with real-world performance

Benchmarks

- Often inspired by real-world problems
- Sometimes even modeled real-world problems
- Problems with interesting properties to test performance of algorithms heuristics
- Doesn't have to correlate with real-world performance

...this is not the class with robots.

Problem definitions

There are many different types of planning...
In this part of the course → classical planning

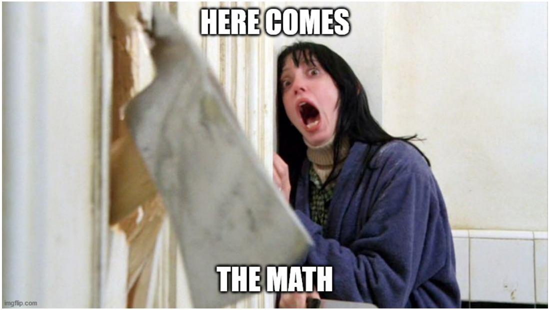
- Fully defined environment
- Deterministic actions
- Domain-independence

Problem definitions

- **STRIPS**
- **FDR**
- **Specify** the model, capture the **structure** of the problem

Problem definitions

- **STRIPS**
- **FDR**
- **Specify** the model, capture the **structure** of the problem



Problem in STRIPS

STRIPS problem $\Pi = \langle F, O, s_{init}, s_{goal}, c \rangle$

- $F = \{f_1, f_2, \ldots f_n\}$ *(facts)*
- $O = \{o_1, o_2, \ldots o_m\}$ *(operators)*
- $s_{init} \subseteq F$ *(initial state)*
- $s_{goal} \subseteq F$ *(goal state specification)*
- $c(o_i) \in \mathbb{R}^+$ *(cost function)*

Problem in STRIPS

STRIPS problem $\Pi = \langle F, O, s_{init}, s_{goal}, c \rangle$

- $F = \{f_1, f_2, \dots f_n\}$ *(facts)*
- $O = \{o_1, o_2, \dots o_m\}$ *(operators)*
- $s_{init} \subseteq F$ *(initial state)*
- $s_{goal} \subseteq F$ *(goal state specification)*
- $c(o_i) \in \mathbb{R}^+$ *(cost function)*

STRIPS operator $o = \langle pre(o), add(o), del(o) \rangle$

- $pre(o) \subseteq F$ *(set of preconditions)*
- $add(o) \subseteq F$ *(set of add effects)*
- $del(o) \subseteq F$ *(set of delete effects)*

Problem in STRIPS

STRIPS problem $\Pi = \langle F, O, s_{init}, s_{goal}, c \rangle$

- $F = \{f_1, f_2, \dots f_n\}$ *(facts)*
- $O = \{o_1, o_2, \dots o_m\}$ *(operators)*
- $s_{init} \subseteq F$ *(initial state)*
- $s_{goal} \subseteq F$ *(goal state specification)*
- $c(o_i) \in \mathbb{R}^+$ *(cost function)*

STRIPS operator $o = \langle pre(o), add(o), del(o) \rangle$

- $pre(o) \subseteq F$ *(set of preconditions)*
- $add(o) \subseteq F$ *(set of add effects)*
- $del(o) \subseteq F$ *(set of delete effects)*
- operators are **well-formed**
  - $add(o) \cap del(o) = \emptyset$
  - $pre(o) \cap add(o) = \emptyset$

Problem in STRIPS

Applicable operator

Operator o is applicable in state s if $pre(o) \subseteq s$.
Resulting state $res(o, s) = (s \setminus del(o)) \cup add(o)$.
State s is a **goal state** iff $s_{goal} \subseteq s$.

Problem in STRIPS

Applicable operator

Operator o is applicable in state s if $pre(o) \subseteq s$.
Resulting state $res(o, s) = (s \setminus del(o)) \cup add(o)$.
State s is a **goal state** iff $s_{goal} \subseteq s$.

Sequence of applicable operators

Sequence of operators $\pi = \langle o_1, o_2, \ldots o_n \rangle$ is applicable in state s_0 if there are states $s_1, s_2, \ldots s_n$ such that o_i is applicable in s_{i-1} and $s_i = res(o_i, s_{i-1})$ for $1 \leq i \leq n$.

- $res(\pi, s_0) = s_n$ *(result of the applied operator sequence π)*
- $c(\pi) = \sum_{o \in \pi} c(o)$ *(cost of applying the operator sequence π)*

Problem in STRIPS

Applicable operator

Operator o is applicable in state s if $pre(o) \subseteq s$.
Resulting state $res(o, s) = (s \setminus del(o)) \cup add(o)$.
State s is a **goal state** iff $s_{goal} \subseteq s$.

Sequence of applicable operators

Sequence of operators $\pi = \langle o_1, o_2, \dots o_n \rangle$ is applicable in state s_0 if there are states $s_1, s_2, \dots s_n$ such that o_i is applicable in s_{i-1} and $s_i = res(o_i, s_{i-1})$ for $1 \leq i \leq n$.

- $res(\pi, s_0) = s_n$ *(result of the applied operator sequence π)*
- $c(\pi) = \sum_{o \in \pi} c(o)$ *(cost of applying the operator sequence π)*

Sequence π is called a **plan** if $s_{goal} \subseteq res(\pi, s_{init})$.

- π is an **optimal plan** is $c(\pi)$ is the minimal cost over all plans

Problem in STRIPS

Reachable state

State s is **reachable** if there exists an applicable sequence of operators π such that $res(\pi, s_{init} = s)$.
Set of all reachable states is denoted $\mathcal{R}_{\Pi}$.

Problem definitions

Problem in FDR

FDR problem $P = \langle \mathcal{V}, \mathcal{O}, s_{init}, s_{goal}, c \rangle$

- $\mathcal{V} = \{V_1, V_2, \dots V_n\}$ *(finite set of variables)*
- $\mathcal{O} = \{o_1, o_2, \dots o_m\}$ *(set of operators)*
- s_{init} *(initial state)*
- s_{goal} *(goal state)*
- $c(o_i) \in \mathbb{R}^+$

Problem in FDR

FDR problem $P = \langle \mathcal{V}, \mathcal{O}, s_{init}, s_{goal}, c \rangle$

- $\mathcal{V}$ *(finite set of variables)*
  - $V \in \mathcal{V}$ *(variable)*
  - D_V *(finite domain of variable V)*
- s *(state)* is partial variable assignment over $\mathcal{V}$
  - vars(s) $= V \in \mathcal{V}$ assigned in s
  - $s[V] =$ value of V in s
  - s is **consistent** with s' if $s[V] = s'[V]$ for all $V \in vars(s')$
  - atom $V = v$ is true in s if $s[V] = v$

Problem in FDR

FDR operator $o = \langle pre(o), \mathit{eff}(o) \rangle$

- $\mathcal{O}$ *(set of operators)*
  - $pre(o)$ = partial assignment over $\mathcal{V}$ *(preconditions)*
  - $\mathit{eff}(o)$ = partial assignment over $\mathcal{V}$ *(effects)*
  - $V = v$ cannot be in both pre(o) and eff(o)

Problem in FDR

Applicable operator

Operator o is applicable in state s if $pre(o)$ is **consistent** with s.

Resulting state $res(o, s) = \begin{cases} eff(o)[V], & \text{if } V \in vars(eff(o))) \\ s[V], & \text{otherwise} \end{cases}$

Problem in FDR

Applicable operator

Operator o is applicable in state s if $pre(o)$ is **consistent** with s.

Resulting state $res(o, s) = \begin{cases} eff(o)[V], & \text{if } V \in vars(eff(o)) \\ s[V], & \text{otherwise} \end{cases}$

Sequence of applicable operators

Sequence of operators $\pi = \langle o_1, o_2, \dots o_n \rangle$ is applicable in state s_0 if there are state $s_1, s_2, \dots s_n$ such that o_i is applicable in s_{i-1} and $s_i = res(o_i, s_{i-1})$ for $1 \leq i \leq n$.

- $res(\pi, s_0) = s_n$ *(result of the applied operator sequence π)*
- $c(\pi) = \sum_{o \in \pi} c(o)$ *(cost of applying the operator sequence π)*

Problem in FDR

Applicable operator

Operator o is applicable in state s if $pre(o)$ is **consistent** with s.

Resulting state $res(o, s) = \begin{cases} eff(o)[V], & \text{if } V \in vars(eff(o))) \\ s[V], & \text{otherwise} \end{cases}$

Sequence of applicable operators

Sequence of operators $\pi = \langle o_1, o_2, \ldots o_n \rangle$ is applicable in state s_0 if there are state $s_1, s_2, \ldots s_n$ such that o_i is applicable in s_{i-1} and $s_i = res(o_i, s_{i-1})$ for $1 \leq i \leq n$.

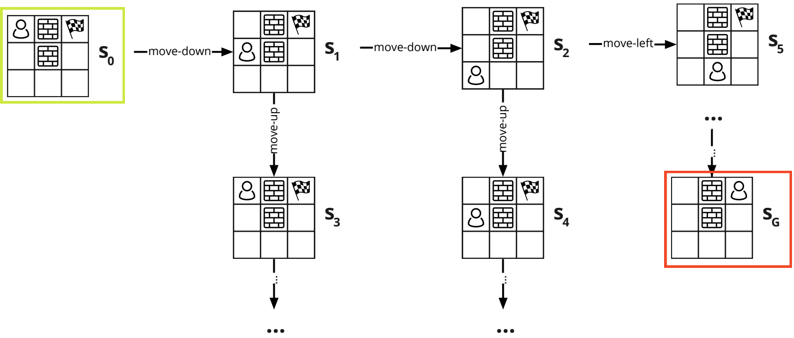
- $res(\pi, s_0) = s_n$ *(result of the applied operator sequence π)*
- $c(\pi) = \sum_{o \in \pi} c(o)$ *(cost of applying the operator sequence π)*

Sequence π is called a **plan** if $res(\pi, s_{init})$ is consistent with s_{goal}.

- π is an **optimal plan** is $c(\pi)$ is the minimal cost over all plans

Transition system

- Both STRIP and FDR have a notion of **state** and **operator**
  - s_0 is the initial state which gets expanded by using $o \in O$ creating new state s' $\rightarrow$ **transition system**



Heuristic functions

- Exhaustive search can be used but can be very slow
- Improvement? **Heuristic function!**
  - Additional information
  - Gives plan cost estimate for a state
  - Helps to navigate the search
  - **Informed search**

Heuristic functions

- Exhaustive search can be used but can be very slow
- Improvement? **Heuristic function!**
  - Additional information
  - Gives plan cost estimate for a state
  - Helps to navigate the search
  - **Informed search**

s-plan

Let Π denote a STRIPS planning task. Sequence of operators π is an **s-plan** iff π is applicable in s and $res(\pi, s)$ is a goal state.
Heuristic $h : \mathcal{R}_\Pi \mapsto \mathbb{R} \cup \{\infty\}$ estimates costs of optimal s-plans.

Heuristic functions

Heuristic function properties

- h^* **optimal** heuristic
- h is **admissible** iff $h(s) \leq h^*(s)$ for every $s \in \mathcal{R}_\Pi$
- h is **goal-aware** iff $h(s) \leq 0$ for every reachable goal state s
- h is **safe** iff $h(s) = \infty$ implies $h^* = \infty$ *(there's no plan)*
- h is **consistent** iff $h(s) \leq h(res(o, s)) + c(o)$ for all reachable states $s \in \mathcal{R}_\Pi$ and $o \in \mathcal{O}$ applicable in s

Exercise

Which statement holds?

- If h is both goal-aware and save, then h is admissible.
- If h is both goal-aware and consistent, then h is admissible.
- If h is both safe and consistent, then h is admissible.

Recap

Goals for today

- think that planning is useful

Recap

Goals for today

- think that planning is useful ✓

Recap

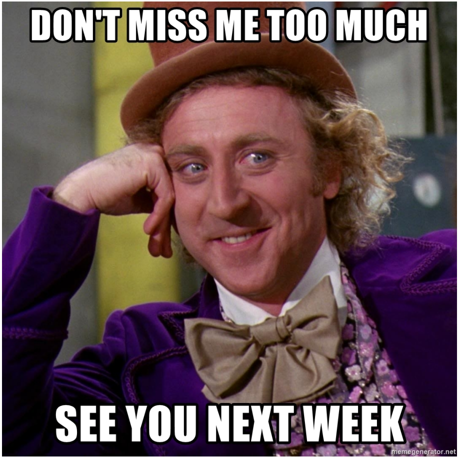
Goals for today

- think that planning is useful ✓
- know STRIPS and FDR problem definition
- know what is a plan
- know what is a heuristic and why we need it
- know heuristic function properties

For more details and structure check out

- Notes on Classical Planning by Daniel Fiser

The End



Feedback form