Relaxation heuristics

Michaela Urbanovská

PUI Tutorial
Week 3

Lecture check

- Any questions regarding the lecture?

Feedback

- THANK YOU for filling the feedback form
- mutexes - will be in more lectures
- speeeeeed - always let me know (tell me, use chat)
- more STRIPS/FDR/PDDL examples - will do!

Obtaining heuristics

- Many different possible ways to obtain a heuristic in general
- General idea:

Obtaining heuristics

- Many different possible ways to obtain a heuristic in general
- General idea: solve a **simplified** version of the problem

Obtaining heuristics

- Many different possible ways to obtain a heuristic in general
- General idea: solve a **simplified** version of the problem
  - relaxation

Obtaining heuristics

- Many different possible ways to obtain a heuristic in general
- General idea: solve a **simplified** version of the problem
  - relaxation
  - abstraction

Obtaining heuristics

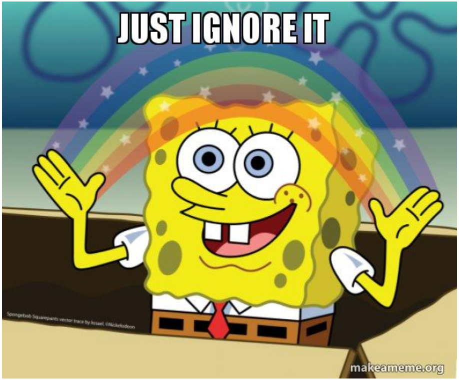
- Many different possible ways to obtain a heuristic in general
- General idea: solve a **simplified** version of the problem
  - relaxation
  - abstraction
- This week: **relaxation**

Relaxation heuristic

- Relaxation is
  - general design technique
  - usually ignoring something
  - simplifying the problem

Relaxation heuristic

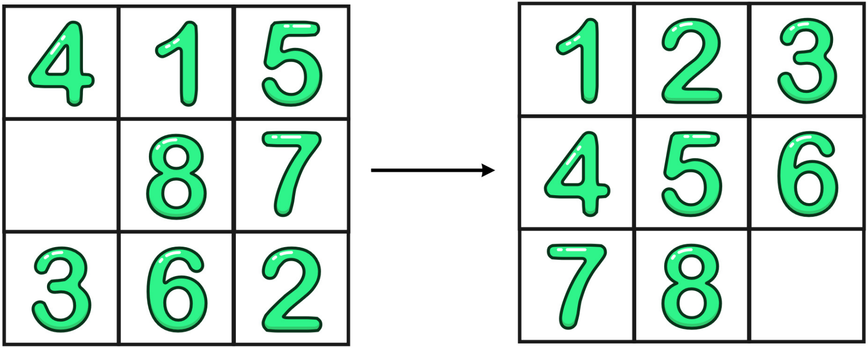
- Relaxation is
  - general design technique
  - usually ignoring something
  - simplifying the problem



Relaxation heuristic

- **Example:** 8-puzzle in STRIPS

- **Example:** 8-puzzle in STRIPS



- How would you formulate it?

Relaxation heuristic - 8-puzzle STRIPS

STRIPS = <F, O, S_i, g>

F = Facts

n1p1 - *number 1 on position 1*
n1p2 - *number 1 on position 2*
…
n8p9 - *number 8 on position 9*
free(p1) - *position 1 is free*
…
free(p9) - *position 8 is free*
next(p1,p2) - *position 1 and 2 are adjacent*
…
next(p8,p9) - *position 8 and 9 are adjacent*

O = Operators <pre, add, del>

move-n1-p1-p2 *(move number 1 from position 1 to position 2)*

- *preconditions:* {n1p1, free(p2)}
- *add effects:* {n1p2, free(p1)}
- *delete effects:* {n1p1}

…

S_i (initial state) = {n1p2, n2p9,n3p7,n4p1,n5p3,n6p8,n7p6,n8p5}
g (goal state specification) = {n1p1,n2p2,n3p3,n4p4,n5p5,n6p6,n7p7,n8p8}

Relaxation heuristic

- What do we relax?

Relaxation heuristic

- What do we relax? Delete effects

Relaxation heuristic

- What do we relax? Delete effects
- **Delete relaxation**

Relaxed STRIPS planning task

Relaxation of a STRIPS planning task $\Pi = \langle F, O, s_{init}, s_{goal}, c \rangle$ is the planning task $\Pi^+ = \langle F, O^+, s_{init}, s_{goal}, c \rangle$ which contains set of relaxed operators.

Relaxation heuristic

- What do we relax? Delete effects
- **Delete relaxation**

Relaxed STRIPS planning task

Relaxation of a STRIPS planning task $\Pi = \langle F, O, s_{init}, s_{goal}, c \rangle$ is the planning task $\Pi^+ = \langle F, O^+, s_{init}, s_{goal}, c \rangle$ which contains set of relaxed operators.

Relaxation of operators

Relaxation of operator $o = \langle pre(o), add(o), del(o) \rangle$ is operator $o^+ = \langle pre(o), add(o), \emptyset \rangle$.

- Let's try it out! http://editor.planning.domains

Relaxation heuristic

- Operator $move(n, p1, p2)$
  - h^* - works with STRIPS definition Π
  - h^+ - works with relaxed STRIPS definition Π^+

h^+ heuristic

The h^+ heuristic computes length of the optimal relaxed plan π^+ which is an optimal solution to the relaxed problem Π^+.

- Is h^+ admissible?

Relaxation heuristic

- Operator $move(n, p1, p2)$
  - h^* - works with STRIPS definition Π
  - h^+ - works with relaxed STRIPS definition Π^+

h^+ heuristic

The h^+ heuristic computes length of the optimal relaxed plan π^+ which is an optimal solution to the relaxed problem Π^+.

- Is h^+ admissible? Yes, if it's optimal.

Relaxation heuristic

- Operator $move(n, p1, p2)$
  - h^* - works with STRIPS definition Π
  - h^+ - works with relaxed STRIPS definition Π^+

h^+ heuristic

The h^+ heuristic computes length of the optimal relaxed plan π^+ which is an optimal solution to the relaxed problem Π^+.

- Is h^+ admissible? Yes, if it's optimal.
- Computing h^+ is still very hard though.

Relaxation heuristic

- Operator $move(n, p1, p2)$
  - h^* - works with STRIPS definition Π
  - h^+ - works with relaxed STRIPS definition Π^+

h^+ heuristic

The h^+ heuristic computes length of the optimal relaxed plan π^+ which is an optimal solution to the relaxed problem Π^+.

- Is h^+ admissible? Yes, if it's optimal.
- Computing h^+ is still very hard though.
- We can compute an **estimate** of h^+.
  - h^{add}
  - h^{max}

h^{add} heuristic

h^{add} heuristic

- STRIPS planning task $\Pi = \langle F, O, s_{init}, s_{goal}, c \rangle$
- $h^{add}(s)$ gives estimate of the distance from s to a state that satisfies s_{goal}
- $h^{add}(s) = \sum_{f \in s} \Delta_0(s, f)$, where
  - $\Delta_0(s, o) = \sum_{f \in pre(o)} \Delta_0(s, f), \forall o \in O$
  - $\Delta_0(s, f) =$
    $$\begin{cases} 0 & \text{if } f \in s, \\ \text{inf} & \text{if } \forall o \in O : f \notin add(o), \\ min\{c(o) + \Delta_0(s, o) | o \in O, f \in add(o)\} & otherwise. \end{cases}$$

h^{max} heuristic

h^{max} heuristic

- STRIPS planning task $\Pi = \langle F, O, s_{init}, s_{goal}, c \rangle$
- $h^{max}(s)$ gives estimate of the distance from s to a state that satisfies s_{goal}
- $h^{max}(s) = max_{f \in s_{goal}} \Delta_1(s, f)$, where
  - $\Delta_1(s, o) = max_{f \in pre(o)} \Delta_1(s, f), \forall o \in O$
  - $\Delta_1(s, f) =$

$$\begin{cases} 0 & \text{if } f \in s, \\ \text{inf} & \text{if } \forall o \in O : f \notin add(o), \\ min\{c(o) + \Delta_1(s, o) | o \in O, f \in add(o)\} & otherwise. \end{cases}$$

Exercise h^{add}, h^{max}

Compute $h^{max}(s_{init})$ for the following problem $\Pi = \langle F, O, s_{init}, s_{goal}, c \rangle$:

$F = \{a, b, c, d, e, f, g\}$

$O =$

	pre	add	del	c
o_1	{a}	{c,d}	{a}	1
o_2	{a,b}	{e}	∅	1
o_3	{b,e}	{d,f}	{a,e}	1
o_4	{b}	{a}	∅	1
o_5	{d,e}	{g}	{e}	1

$s_{init} = \{a, b\}$ $s_{goal} = \{f, g\}$

h^{max} algorithm

Algorithm 1: Algorithm for computing $\mathrm{h}^{\max}(s)$.

Input: $\Pi = \langle \mathcal{F}, \mathcal{O}, s_{init}, s_{goal}, \mathrm{c} \rangle$, state s
Output: $\mathrm{h}^{\max}(s)$

for each $f \in s$ **do** $\Delta_1(s,f) \leftarrow 0$;
for each $f \in \mathcal{F} \setminus s$ **do** $\Delta_1(s,f) \leftarrow \infty$;
for each $o \in \mathcal{O}, \mathit{pre}(o) = \emptyset$ **do**
    for each $f \in \mathit{add}(o)$ **do** $\Delta_1(s,f) \leftarrow \min\{\Delta_1(s,f), \mathrm{c}(o)\}$;
end
for each $o \in \mathcal{O}$ **do** $U(o) \leftarrow |\mathrm{pre}(o)|$;
$C \leftarrow \emptyset$;
while $s_{goal} \not\subseteq C$ **do**
    $k \leftarrow \arg\min_{f \in \mathcal{F} \setminus C} \Delta_1(s,f)$;
    $C \leftarrow C \cup \{k\}$;
    for each $o \in \mathcal{O}, k \in \mathit{pre}(o)$ **do**
        $U(o) \leftarrow U(o) - 1$;
        if $U(o) = 0$ **then**
            for each $f \in \mathit{add}(o)$ **do**
                $\Delta_1(s,f) \leftarrow \min\{\Delta_1(s,f), \mathrm{c}(o) + \Delta_1(s,k)\}$;
            end
        end
    end
end
$\mathrm{h}^{\max}(s) = \max_{f \in s_{goal}} \Delta_1(s,f)$;

Exercise h^{add}, h^{max}

Compute $h^{add}(s_{init})$ for the following problem $\Pi = \langle F, O, s_{init}, s_{goal}, c \rangle$:

$F = \{a, b, c, d, e, f, g\}$

$O =$

	pre	add	del	c
o_1	{a}	{c,d}	{a}	1
o_2	{a,b}	{e}	∅	1
o_3	{b,e}	{d,f}	{a,e}	1
o_4	{b}	{a}	∅	1
o_5	{d,e}	{g}	{e}	1

$s_{init} = \{a, b\}$ $s_{goal} = \{f, g\}$

What's the difference in the algorithm?

Heuristic properties

Heuristic dominance

Admissible heuristic h_1 dominates an admissible heuristic h_2 if for every state s $h_1(s) \geq h_2(s)$

- h^+ is **admissible**, **consistent**
- h^{max} is **admissible**, **consistent**
- h^{add} is **not admissible**, **nor consistent** but can be very informative
- $h^{max} \leq h^+ \leq h^*$

Recap

- Relaxation heuristics
- Delete relaxation
- h^+, h^{max} and h^{add} heuristics
- Implementation of h^{max}

The End

Feedback form link