Course on Automated Planning: Planning as Heuristic Search

Hector Geffner
ICREA & Universitat Pompeu Fabra
Barcelona, Spain

From Strips Problem P to State Model $S(P)$

A Strips problem $P = \langle F, O, I, G \rangle$ determines **state model** $S(P)$ where

- the states $s \in S$ are **collections of atoms** from F
- the initial state s_0 is I
- the goal states s are such that $G \subseteq s$
- the actions a in $A(s)$ are ops in O s.t. $Pre(a) \subseteq s$
- the next state is $s' = s - Del(a) + Add(a)$
- action costs $c(a, s)$ are all 1

How to solve $S(P)$?

Heuristic Search Planning

- Explicitly **searches** graph associated with model $S(P)$ with **heuristic** $h(s)$ that estimates cost from s to goal

- **Key idea:** Heuristic h extracted **automatically** from problem P

This is the mainstream approach in classical planning (and other forms of planning as well), enabling the solution of problems over **huge spaces**

Heuristics for Classical Planning

- Key development in planning in the 90's, is automatic extraction of **heuristic functions** to guide search for plans

- The general idea was known: heuristics often **explained** as **optimal** cost functions of **relaxed** (simplified) problems (Minsky 61; Pearl 83)

- Most common relaxation in planning, P^+, obtained by dropping **delete-lists** from ops in P. If $c^*(P)$ is optimal cost of P, then

$$h^+(P) \stackrel{\text{def}}{=} c^*(P^+)$$

- Heuristic h^+ **intractable** but easy to **approximate**; i.e.
  - ▷ *computing* **optimal** *plan for P^+ is* **intractable**, but
  - ▷ *computing a non-optimal plan for P^+* (**relaxed plan**) *easy*

- State-of-the-art heuristics as in FF or LAMA still rely on P^+ . . .

Additive Heuristic

- For all **atoms** p:

$$h(p; s) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } p \in s \text{, else} \\ \min_{a \in O(p)} [cost(a) + h(Pre(a); s)] \end{cases}$$

- For **sets** of atoms C, assume **independence**:

$$h(C; s) \stackrel{\text{def}}{=} \sum_{r \in C} h(r; s)$$

- Resulting **heuristic function** $h_{add}(s)$:

$$h_{add}(s) \stackrel{\text{def}}{=} h(Goals; s)$$

Heuristic not admissible but informative and fast

Max Heuristic

- For all **atoms** p:

$$h(p;s) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } p \in s \text{, else} \\ \min_{a \in O(p)} [1 + h(Pre(a);s)] \end{cases}$$

- For **sets** of atoms C, replace **sum** by **max**

$$h(C;s) \stackrel{\text{def}}{=} max_{r \in C} h(r;s)$$

- Resulting **heuristic function** $h_{max}(s)$:

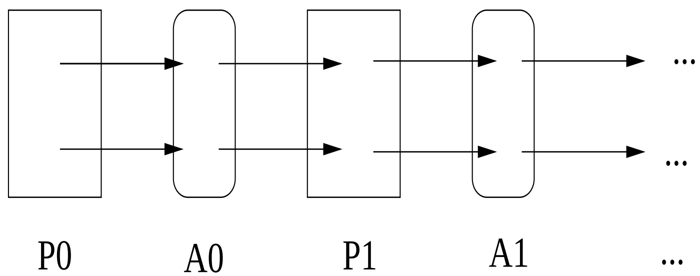
$$h_{max}(s) \stackrel{\text{def}}{=} h(Goals;s)$$

Heuristic admissible but not very informative . . .

Max Heuristic and (Relaxed) Planning Graph

- Build reachability graph P_0, A_0, P_1, A_1, . . .

$$
\begin{aligned}
P_0 &= \{p \in s\} \\
A_i &= \{a \in O \mid Pre(a) \subseteq P_i\} \\
P_{i+1} &= P_i \cup \{p \in Add(a) \mid a \in A_i\}
\end{aligned}
$$



– Graph implicitly **represents** max heuristic:

$$h_{max}(s) = \text{min } i \text{ such that } G \subseteq P_i$$

Heuristics, Relaxed Plans, and FF

- (Relaxed) Plans for P^+ can be obtained from **additive** or **max** heuristics by recursively collecting **best supports** backwards from goal, where a_p is **best support** for p in s if

$$a_p = \text{argmin}_{a \in O(p)} h(a_p) = \text{argmin}_{a \in O(p)} [1 + h(Pre(a))]$$

- A plan $\pi(p; s)$ for p in delete-relaxation can then be computed backwards as

$$\pi(p; s) = \begin{cases} \emptyset & \text{if } p \in s \\ \{a_p\} \cup \cup_{q \in Pre(a_p)} \pi(q; s) & \text{otherwise} \end{cases}$$

- The **relaxed plan** $\pi(s)$ for the **goals** obtained by **planner FF** using $h = h_{max}$

- More accurate h obtained then from **relaxed plan** π as

$$h(s) = \sum_{a \in \pi(s)} cost(a)$$

Variations in state-of-the-art Planners: EHC, Helpful Actions, Landmarks

- In original formulation of **planning as heuristic search**, the states s and the heuristics $h(s)$ become **black boxes** used in **standard search algorithms**

- More recent planners like **FF** and **LAMA** go beyond this in two ways

- They exploit the structure of the heuristic and/or problem further:
  - ▷ **Helpful Actions**
  - ▷ **Landmarks**

- They use novel search algorithms
  - ▷ **Enforced Hill Climbing (EHC)**
  - ▷ **Multi-queue Best First Search**

- The result is that they can often solve **huge problems**, **very fast**. Not always though; try them!