Planning, Execution & Learning: Heuristic Search Planning I

Manuela Veloso
Reid Simmons

Search

- Uninformed
  - Depth-first
  - Breadth-first

- Informed (Heuristic)
  - Hill-climbing
  - Best-first
  - A*
Introduce a function \( h(s) \) to estimate the unknown distance from state \( s \) to the goal. Our best guess is that \( A \) is closer to \( GOAL \) than \( B \) so maybe it is a more promising state to expand.

\( h(A) = 3 \)
\( h(B) = 6 \)
\( h(B) = 10 \)

**Heuristics – Correct or Incorrect**

- \( h(s) = \) Euclidean distance to \( GOAL \) – an heuristic.
**Heuristic Search Strategies**

- **Hill-Climbing**
  - Rationale: Heuristics tend to be better discriminators amongst local alternatives than as global (absolute) estimate
  - Random “restarts” when stuck

- **Best-First** \( f(s) = h(s) \)

- **A\(^*\)** \( f(s) = g(s) + h(s) \)

- **Weighted A\(^*\)**
  - \( H(s) = \text{cost-so-far}(s) + W \times \text{estimated-cost}(s) \)
  - Not admissible, but tends to perform much better than A\(^*\)

---

**Search Heuristics**

- **Admissible**
  - *What?*
  - *Why Important?*

- **Informed**
  - *What?*
  - *Why Important?*
**Heuristic Search Planning**

- **Basic Idea**
  - *Automatically Analyze Domain/Problems to Derive Heuristic Estimates to Guide Search*

- **Decisions**
  - How to evaluate search states
  - How to use the evaluations to guide search
  - How to choose which part of plan to work on next

- **Resurgence in Total-Order, State-Space Planners**
  - Best such planners (FF, FD, LAMA) dominate other types
  - Still an ongoing topic for research

**How to Generate Heuristics**

- **Domain-Specific**
  - Program in (or *learn*) heuristics specifically for that domain

- **Domain Analysis**
  - Preprocess domain to generate meta-control knowledge

- **On-Line**
  - Solve a *relaxed* form of the problem
  - Use as estimate for original problem
Types of On-Line Heuristics

• Assume complete subgoal independence

• Assume no negative interactions

• Assume limited negative interactions

HSP (Bonet & Geffner, 1997)

• Heuristic State-Space Planner
  – Can do either progression or regression

• Relax Problem by Eliminating “Delete” Lists
  – Essentially compute transitive closure of actions, starting at initial state
    • “Reachability” analysis
  – Cost of literal is stage/level at which first appears
  – Continue until no new literals are added
  – Similar to GraphPlan’s forward search
Computing Costs of Literals

- **On(C, A)**
- **On(A, Table)**
- **On(B, Table)**
- **Handempty**
- **Clear(C)**
- **Clear(B)**

- **Pick(C, A)**
- **PickT(B)**
- **1 Holding(C)**
- **1 Holding(B)**
- **1 Clear(A)**

- **PutT(C)**
- **Put(C, A)**
- **Put(C, B)**
- **PutT(B)**
- **Put(B, A)**
- **Put(B, C)**
- **PickT(A)**

- **2 On(C, Table)**
- **2 On(C, B)**
- **2 On(B, A)**
- **2 On(B, C)**
- **2 Holding(A)**

- **PickT(C)**
- **Pick(C, B)**
- **Pick(B, A)**
- **Pick(B, C)**
- **Put(A, B)**
- **Put(A, C)**

- **3 On(A, B)**
- **3 On(A, C)**

- **On(A, B) & On(B, C)**  **Estimate:** 5  **Actually:** 6
- **On(A, C) & On(A, B)**  **Estimate:** 5  **Actually:** 4

**HSP Heuristics**

- **Max**
  - Cost of action is *maximum* over costs of preconditions
  - Focus on cost of *most difficult* goals
  - Admissible, but not very informed

- **Sum**
  - Cost of action is *sum* of precondition costs
  - Combines *all* goals
  - Informed, but not admissible

- **H^2**
  - Solve for *pairs* of literals
  - Take maximum cost over all pairs
  - Informed, and claimed to be admissible
**Computing and Estimating Cost**

- Cost of achieving an atom \( p \) from state \( s \):
  
  \[
g_s(p) = \begin{cases} 
  0 & \text{if } p \in s \\
  \min_{o \in O(s)} [1 + g_s(Precc(op))] & \text{otherwise}
\end{cases}
\]

- Computation of cost by Forward Chaining:
  - Initialize
    
    \( g_s(p) \) are initialized to 0 if \( p \in s \) and to \( \infty \) otherwise
  - Update – operator applied to state adds \( p \)
    
    \[
g_s(p) := \min \left( g_s(p), 1 + g_s(Precc(op)) \right)
    \]
  - Continue until no change.

**Heuristic Value of State \( s \)**

- Estimate of achieving \( G \) from a state \( s \):
  
  \( h(s) \overset{\text{def}}{=} g_s(G) \)

- Cost of achieving a SET of atoms
  
  \( g_s(G), g_s(Precc(op)) \)

- Additive heuristic
  
  \[
g_s^+(C) = \sum_{r \in C} g_s(r) \quad \text{(additive costs)}
\]

- Max heuristic
  
  \[
g_s^{max}(C) = \max_{r \in C} g_s(r) \quad \text{(max costs)}
\]
**HSP Planner**

- Uses the additive heuristic
- Very simple basic hill-climbing search
  - “At every step, one of the best children (minimize heuristic) is selected for expansion and the same process is reached until a goal is achieved. Ties are broken randomly.”
- Extended hill-climbing search
  - Counting number of steps without decrementing heuristic value
  - Random restart after some threshold without progress

**HSP versus FF Heuristics**

<table>
<thead>
<tr>
<th>name</th>
<th>(pre, add, del)</th>
</tr>
</thead>
<tbody>
<tr>
<td>opG₁</td>
<td>({P}, {G₁}, ∅)</td>
</tr>
<tr>
<td>opG₂</td>
<td>({P}, {G₂}, ∅)</td>
</tr>
<tr>
<td>opP</td>
<td>(∅, {P}, ∅)</td>
</tr>
</tbody>
</table>

- HSP additive heuristic: P value 1, G1 and G2 value 2, estimate distance from initial state (empty) to goal state (G1 and G2) is 4. But just three steps: opP, opG₁, opG₂
**FF (Hoffmann, 2000)**

- FF (Fast Forward) refines HSP heuristic

- Takes *positive* interactions into account
  - Avoids double-counting of actions

- Similar to *GraphPlan’s* forward search combined with a *relaxed* version of its backward search
  - Ignores negative interactions

- Admissible and Informed

**FF State Evaluation Heuristic**

<table>
<thead>
<tr>
<th>State</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>On(A, C)</td>
<td>Pick(C, A)</td>
</tr>
<tr>
<td>On(A, Table)</td>
<td>Handempty</td>
</tr>
<tr>
<td>On(B, Table)</td>
<td>Clear(B)</td>
</tr>
<tr>
<td>PickT(B)</td>
<td></td>
</tr>
<tr>
<td>On(C, A)</td>
<td>Hold(C)</td>
</tr>
<tr>
<td>On(A, Table)</td>
<td>Handempty</td>
</tr>
<tr>
<td>On(B, Table)</td>
<td>Clear(B)</td>
</tr>
<tr>
<td>Handempty</td>
<td></td>
</tr>
<tr>
<td>Clear(C)</td>
<td></td>
</tr>
<tr>
<td>Clear(B)</td>
<td></td>
</tr>
<tr>
<td>PickT(B)</td>
<td></td>
</tr>
<tr>
<td>PutT(C)</td>
<td></td>
</tr>
<tr>
<td>Put(C, A)</td>
<td></td>
</tr>
<tr>
<td>Put(C, B)</td>
<td></td>
</tr>
<tr>
<td>PutT(B)</td>
<td></td>
</tr>
<tr>
<td>Put(B, A)</td>
<td></td>
</tr>
<tr>
<td>Put(B, C)</td>
<td></td>
</tr>
<tr>
<td>PickT(A)</td>
<td></td>
</tr>
<tr>
<td>On(A, C)</td>
<td></td>
</tr>
<tr>
<td>On(A, Table)</td>
<td></td>
</tr>
<tr>
<td>On(B, Table)</td>
<td></td>
</tr>
<tr>
<td>Handempty</td>
<td></td>
</tr>
<tr>
<td>Clear(C)</td>
<td></td>
</tr>
<tr>
<td>Clear(B)</td>
<td></td>
</tr>
<tr>
<td>Pick(C, A)</td>
<td></td>
</tr>
<tr>
<td>PutT(B)</td>
<td></td>
</tr>
<tr>
<td>Put(C, A)</td>
<td></td>
</tr>
<tr>
<td>Put(C, B)</td>
<td></td>
</tr>
<tr>
<td>PutT(B)</td>
<td></td>
</tr>
<tr>
<td>Put(B, A)</td>
<td></td>
</tr>
<tr>
<td>Put(B, C)</td>
<td></td>
</tr>
<tr>
<td>PickT(A)</td>
<td></td>
</tr>
<tr>
<td>On(A, C)</td>
<td></td>
</tr>
<tr>
<td>On(A, Table)</td>
<td></td>
</tr>
<tr>
<td>On(B, Table)</td>
<td></td>
</tr>
<tr>
<td>Handempty</td>
<td></td>
</tr>
<tr>
<td>Clear(C)</td>
<td></td>
</tr>
<tr>
<td>Clear(B)</td>
<td></td>
</tr>
<tr>
<td>PickT(B)</td>
<td></td>
</tr>
<tr>
<td>On(A, C, B)</td>
<td></td>
</tr>
<tr>
<td>On(A, C) &amp; On(C, B)</td>
<td></td>
</tr>
</tbody>
</table>
“Enforced” Hill Climbing

- Used by FF to Avoid “Wandering” on “Plateaus” or in Local Minima
  - Perform breadth-first search until find some descendant state whose heuristic value is less than the current state

- Shown to be Very Effective
  - Especially when search space is pruned to eliminate actions that are “unlikely” to lead to goal achievement

Discussion

- Progression: Need to Calculate Heuristic Every Step
- Regression: Just Calculate Heuristic Once
- Heuristic Search Using Progression Generally More Robust
- HSP and FF strongly based on state-space search
- HSP and FF Heuristics Outperform Partial-Order Planners in common planning problems
  - Ground actions seem to be the big difference
    - Easier to estimate cost without variables
    - Forward search provides reachability analysis
- Similar Techniques Applicable to Partial-Order Planners
  - REPOP (Nguyen & Kambhampati, 2001)
  - VHPOP (Younes & Simmons, 2001)