Artificial Intelligence Planning

Heuristic Search Planning

A Tiny (But Tasty) Appetizer

Jörg Hoffmann

SAARLAND UNIVERSITY
COMPUTER SCIENCE

Coursera, Jan/Feb 2013

Thanks to everybody who contributed to the work described here!
Agenda

1. Why?
2. What?
3. How?
4. Theory
5. Practice
6. And Now?
Why?

**IPC = The International Planning Competition:**
- **IPC 2000 Winner:** heuristic search.
- **IPC 2002 Winner:** heuristic search.
- **IPC 2004 Winner:** satisficing: heuristic search, optimal: SAT.
- **IPC 2006 Winner:** satisficing: heuristic search, optimal: SAT.
- **IPC 2008 Winner:** satisficing: heuristic search, optimal: symbolic search.
- **IPC 2011 Winner:** satisficing: heuristic search (first 12 places), optimal: heuristic search (first 9 places).

**ATTENTION!**
- This is only for the fully-automatic deterministic tracks of the IPC.
- This does NOT mean heuristic search is universally better; it’s only the IPC setup.
- ”Winner” is a very inadequate summary of such huge and complex events.

→ All I’m saying is: This approach has been mainstream in academic planning research during the last decade, and has produced a lot of interesting results.
→ Heuristic function $h$ maps world states $s$ to an estimate $h(s)$ of goal distance. Search prefers to explore states with small $h$. 
Problem: Find a route from Saarbruecken To Edinburgh.
What (2): Heuristic Functions

Simplified Problem: Throw away the map.
Heuristic function: Straight line distance.
**How?**

- **Ignoring Deletes**
  - $h_{\text{max}}$
  - $h^+$

- **Abstractions**
  - PDB
  - M&S

- **Critical Paths**
  - $h^1$
  - $h^2$
  - $h^3$
  - ...$

- **Landmarks**
  - $h_{LM}^L$

---

Jörg Hoffmann  
Heuristic Search Planning  
A Tiny (But Tasty) Appetizer
Abstractions in the 15-Puzzle

Jörg Hoffmann
Heuristic Search Planning
A Tiny (But Tasty) Appetizer
Abstractions in the 15-Puzzle

\[ \rightarrow h = \text{Solution to Smaller (and Easier) Puzzle} \]
How?

Ignoring Deletes

\[ h_{\text{max}} \]
\[ h^+ \]

Abstractions

PDB
M&S

Critical Paths

\[ h^1 \]
\[ h^2 \]
\[ h^3 \]
\[ \ldots \]

Landmarks

\[ h^{\text{LM}}_L \]
Problem: Bring small key to position 1.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
</table>

Landmarks:
- Lock-open.
- Have-big-key.
- Have-small-key.
- ...

→ $h = \text{"Number of open items on the to-do list"}$
Why
What?
How
Theory
Practice
And Now?
References

How?

Ignoring Deletes

\( h_{\text{max}} \)
\( h^+ \)

Abstractions

PDB
M&S

Critical Paths

\( h^1 \)
\( h^2 \)
\( h^3 \)
\( \ldots \)

Landmarks

\( h_{\text{LM}} \)
\( h_L \)
$\rightarrow h^m = \text{Most Expensive } m\text{-Sub-Tour}$

© 2013 Google
How?

Ignoring Deletes

\( h^{\text{max}} \)
\( h^+ \)

Abstractions

PDB
M&S

Critical Paths

\( h^1 \)
\( h^2 \)
\( h^3 \)
\( \ldots \)

Landmarks

\( h_{LM}^L \)
Ignoring Deletes in TSP

→ $h =$ Minimum Spanning Tree
Theory

Compilability between lower-bound $h$:

Ignoring Deletes

$h^{\text{max}} \leq h^+$

$h^{\text{max}} = h^1$

Abstractions

$PDB \preceq M&S$

$PDB \not\preceq h^m$

Critical Paths

$h^1 \preceq h^2 \preceq h^3 \preceq \ldots$

$h^1 \equiv h^{\text{LM}}$

Landmarks

$h^{\text{LM}}$

$h^1 \equiv h^{\text{LM}}$

$h^1 \not\preceq PDB$

$h^1 \preceq M&S$

$h^m \not\preceq M&S$
Problem: Regular security checks by running (millions of) attacks.

→ Solution@Core Security: Heuristic Search Planning!
If your appetite is stimulated, you can have a look at my lecture slides . . .

http://fai.cs.uni-saarland.de/teaching/winter12-13/planning.html

. . . and/or google some of the great people who contributed to this area:

- Blai Bonet
- Carmel Domshlak
- Hector Geffner
- Patrik Haslum
- Malte Helmert

→ This list is very incomplete, there’s lots more people who contributed!
And Now?

You can also have a look at some papers:

- **Abstractions**: [Edelkamp (2001); Haslum et al. (2007); Helmert et al. (2007)]

- **Landmarks**: [Hoffmann et al. (2004); Karpas and Domshlak (2009); Richter and Westphal (2010)]

- **Critical Paths**: [Haslum and Geffner (2000)]

- **Ignoring deletes**: [Bonet and Geffner (2001); Hoffmann and Nebel (2001); Keyder et al. (2012)]

- **Compilability**: [Helmert and Domshlak (2009)]

- **Security tests**: [Lucangeli et al. (2010); Sarraute et al. (2012)]
References I


Carlos Sarrate, Olivier Buffet, and Jörg Hoffmann. POMDPs make better hackers: Accounting for uncertainty in penetration testing. In Jörg Hoffmann and Bart Selman, editors, *Proceedings of the 26th National Conference of the American Association for Artificial Intelligence (AAAI’12)*, Toronto, ON, Canada, July 2012. AAAI Press.