Classical Planning Heuristics

5. Heuristics for Classical Planning I

Gabriele Röger

ICAPS 2018 Summer School

June 22, 2018

Heuristics

Planning as Heuristic Search

Planning as Heuristic Search

general search algorithm (e.g. A*, greedy best-first search)

$+$ distance estimator (heuristic function)

Desirable Properties of Heuristics for Optimal Planning

heuristic h maps states to numbers in $\mathbb{R}_{\geq 0} \cup \{\infty\}$

Desirable properties for optimal planning:

- admissibility: $h(s) \leq h^*(s)$
  for all states s
- consistency: $h(s) \leq cost(o) + h(s')$
  for all state transitions $s \xrightarrow{o} s'$

h^*: actual distance to goal ("perfect heuristic")

Desirable Properties for All Heuristics

Desirable property for optimal and satisficing planning:

- accuracy: $h(s)$ should be "close" to $h^*(s)$

The Challenge

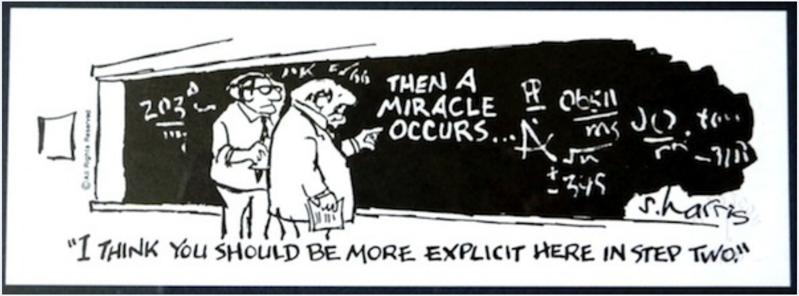
How do we come up with precise estimates
in a domain-independent fashion?

How to Come Up with Good Heuristics

How do we come up with good heuristics?

A commonly held view:

Inspiration



Must we wait for inspiration to strike?

How to Come Up with Heuristics

How do we come up with good heuristics?

Another commonly held view:

Perspiration

"None of my inventions came by accident. I see a worthwhile need to be met and I make trial after trial until it comes. What it boils down to is one per cent inspiration and ninety-nine per cent perspiration."

— Thomas Alva Edison (1929)

Phrased less positively:
"Throw enough mud at the wall, some of it will stick."

How to Come Up with Heuristics

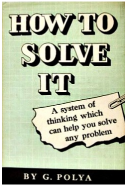
How do we come up with good heuristics?

Our recommendation:

Careful Scientific Study

"First, you have to understand the problem."

— George Pólya (1945)



The Science of Heuristics

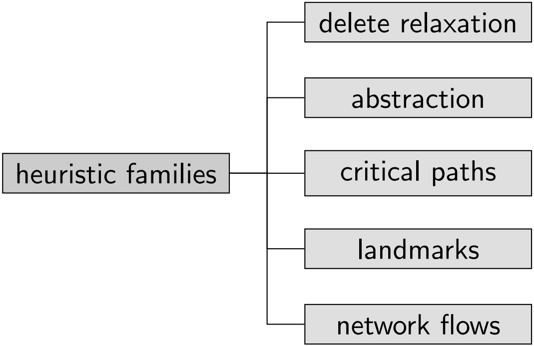
Ask (and answer) questions:

- Dissect existing approaches: where do they work? Where not? Why and why not?
- Can specific approaches be distilled to general ideas? Can general ideas be applied more specifically?
- Compare existing approaches: does one dominate another? What do they have in common? How are they different? Can their strengths be combined?

Five Families of Heuristics

How do we come up with heuristics for general problems?

$\rightsquigarrow$ five major approaches in the literature



Running Example: FreeCell

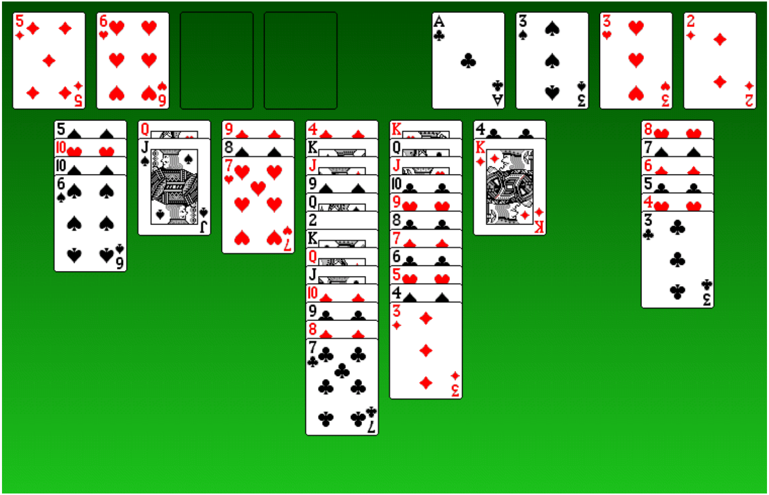


image credits: GNOME Project (GNU General Public License)

Heuristics in Fast Downward

When a heuristic is implemented in Fast Downward,
we mention its plug-in name.

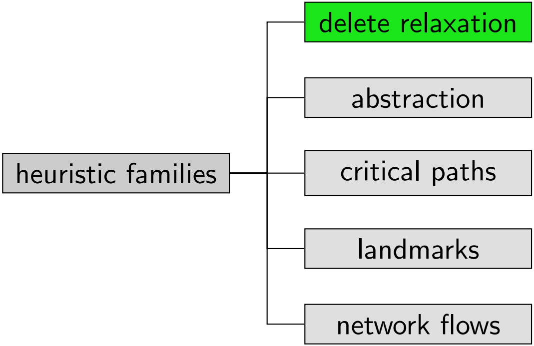
- Many heuristics have options, sometimes with bad defaults.
- We use dots (e.g., `lmcount(...)`) when options matter a lot.

Want to find out more?

- `http://www.fast-downward.org/Doc/Heuristic`
- Check out the cited papers.
- public mailing list
  $\rightsquigarrow$ link on `http://www.fast-downward.org`
- Ask us!

Delete Relaxation

Five Families of Heuristics



Planning Heuristics: Delete Relaxation

Five classes of heuristics:

1. Delete Relaxation

Estimate cost to goal by considering simpler planning task without negative side effects of actions.

Example: Delete Relaxation in FreeCell

Problem constraints dropped by the delete relaxation in FreeCell:

- free cells and free tableau positions remain available after moving cards into them
- cards remain movable and remain valid targets for other cards after moving cards on top of them

Delete Relaxation

delete relaxation: ignore "bad effects" of actions

- What is a bad effect?
- easy for STRIPS: it's always "better for us" if a fact is true!

$\rightsquigarrow$ bad effect = delete effect

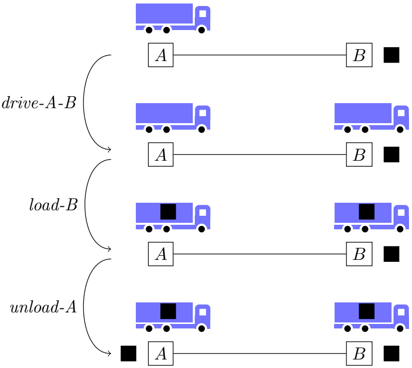
$\rightsquigarrow$ delete relaxation of a task: drop all delete effects

Use delete relaxation as basis for heuristics:

- in each state, estimate cost to the goal in delete relaxation

Delete Relaxation = Accumulating Values

Delete Relaxation = Accumulating Values



Optimal Relaxed Cost

- $h^+(s)$: minimal total cost to reach the goal from s
- NP-hard to compute (Bylander, AIJ 1994)
  or approximate by constant factor (Betz & Helmert, KI 2009)
  - $\rightarrow$ use polynomial-time approximation,
    e.g. FF heuristic uses cost of possibly suboptimal plan

Delete Relaxation Heuristics in the Literature

Delete relaxation approaches in the literature (admissible in red):

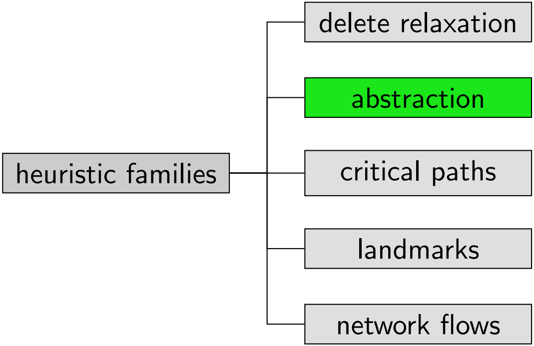
- max heuristic (Bonet & Geffner, ECP 1999; AIJ 2001)
- additive heuristic (Bonet & Geffner, ECP 1999; AIJ 2001)
- FF heuristic, h^+ heuristic (Hoffmann & Nebel, JAIR 2001)
- pairwise max heuristic (Mirkis & Domshlak, ICAPS 2007)
- set-additive heuristic (Keyder & Geffner, ECAI 2008)
- Steiner tree heuristic (Keyder & Geffner, IJCAI 2009)
- landmark-cut heuristic (Helmert & Domshlak, ICAPS 2009)
- red-black heuristic (Domshlak et al., JAIR 2015)
- explicitly represented conjunctions (Haslum, ICAPS 2012; Keyder et al., JAIR 2014)

in Fast Downward

`hmax()`, `add()`, `ff()`, `lmcut()`

Abstraction

Five Families of Heuristics



Planning Heuristics: Abstraction

Five classes of heuristics:

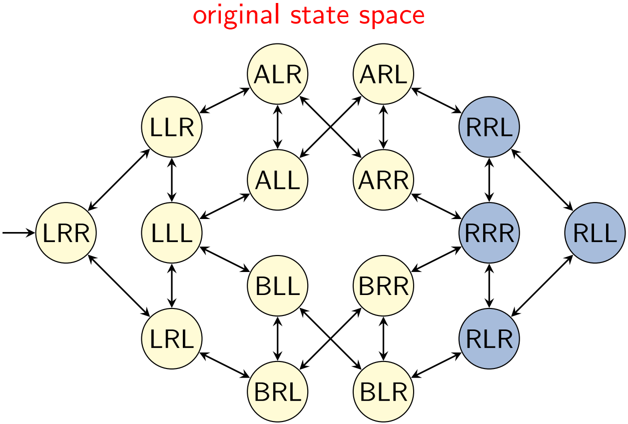
2. Abstraction

Estimate cost by projecting the state space to a smaller space (applying a graph homomorphism).

Example: Abstraction in FreeCell

One possible abstraction for FreeCell:
project away all cards that are not 10s, Js, Qs or Ks.

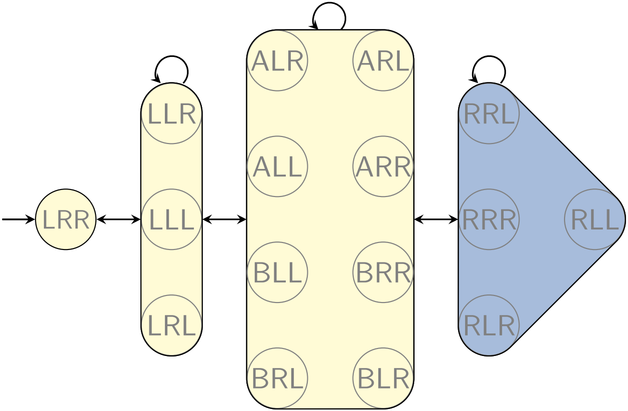
Abstraction: Example



- state variable *package*: $\{L, R, A, B\}$
- state variable *truck A*: $\{L, R\}$
- state variable *truck B*: $\{L, R\}$

Abstraction: Example

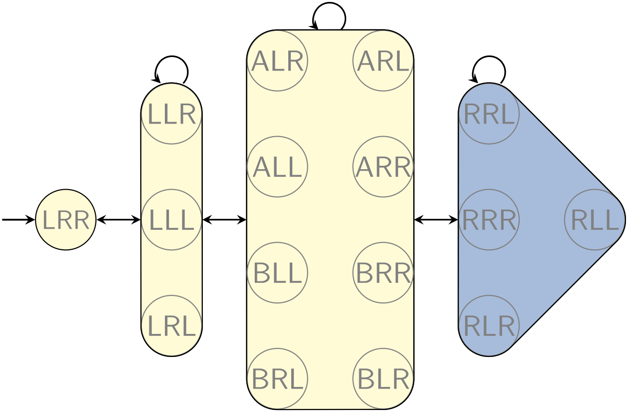
(an) abstract state space



Remark: Most edges correspond to several parallel transitions with different labels.

Abstraction: Example

$$h^{\alpha}(\{p \mapsto \mathsf{L}, t_{\mathsf{A}} \mapsto \mathsf{R}, t_{\mathsf{B}} \mapsto \mathsf{R}\}) = 3$$



Abstraction Heuristics

- Abstract state space is derived from original state space as specified by an abstraction function
- Abstraction function defines which states should be distinguished

Abstraction Heuristics

- Abstract state space is derived from original state space as specified by an abstraction function
- Abstraction function defines which states should be distinguished
- Preserve all original paths in abstract state space
- Do not relax more than required by abstraction function

Induced Abstraction

Definition (induced abstraction)

Let $\mathcal{S} = \langle S, s_0, S_\star, A, cost, T \rangle$ be a state space and let $\alpha : S \to S'$ be a surjective function.

The abstraction of $\mathcal{S}$ induced by α is the state space $\mathcal{S}^\alpha = \langle S', s'_0, S'_\star, A, cost, T' \rangle$ with:

- $s'_0 = \alpha(s_0)$
- $S'_\star = \{\alpha(s) \mid s \in S_\star\}$
- $T' = \{\langle \alpha(s), a, \alpha(t) \rangle \mid \langle s, a, t \rangle \in T\}$

Abstraction Heuristic

Definition (abstraction heuristic)

For state space $\mathcal{S}$ and abstraction function α, the heuristic estimate $h^\alpha(s)$ for state s is the cost of a cheapest path from $\alpha(s)$ to a goal state in $\mathcal{S}^\alpha$.

Abstraction Heuristic

Definition (abstraction heuristic)

For state space $\mathcal{S}$ and abstraction function α, the heuristic estimate $h^\alpha(s)$ for state s is the cost of a cheapest path from $\alpha(s)$ to a goal state in $\mathcal{S}^\alpha$.

Abstraction heuristics are admissible and consistent.

Abstraction Heuristics in the Literature

Abstraction heuristics in the literature (admissible in red):

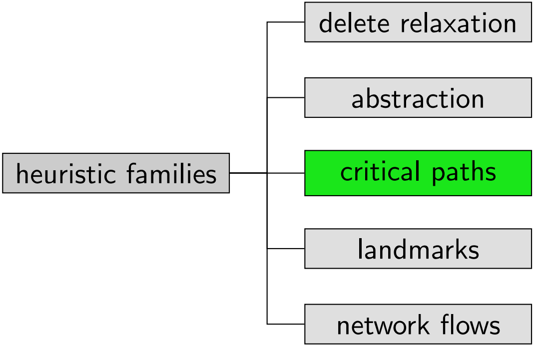
- pattern databases (PDBs) (Edelkamp, ECP 2001; Haslum et al., AAAI 2007; Pommerening et al., IJCAI 2013)
- symbolic PDBs (Edelkamp, AIPS 2002)
- constrained PDBs (Haslum et al., AAAI 2005)
- merge-and-shrink (Dräger et al., SPIN 2006; Helmert et al., ICAPS 2007; Sievers et al., AAAI 2014; Sievers, PhD 2017)
- structural patterns (Katz & Domshlak, ICAPS 2008)
- Cartesian abstraction (Seipp & Helmert, ICAPS 2013; ICAPS 2014; JAIR 2018)

in Fast Downward

`pdb(...)`, `zopdbs(...)`, `cpdbs(...)`, `gapdb(...)`, `ipdb(...)`, `merge_and_shrink(...)`, `cegar(...)`

Critical Paths

Five Families of Heuristics



Planning Heuristics: Critical Paths

Five classes of heuristics:

3. Critical Paths

Estimate cost by critical path length (makespan) of a concurrent solution for a simplified problem.

In the simplification, a set of subgoals is considered reachable when all size-m subsets are reachable; $m \in \mathbb{N}_1$ is a parameter.

Example: Critical Paths in FreeCell

Possible critical path for single subgoals (h^1):

- Solving the FreeCell task requires four subgoals: have each of $\diamondsuit K$, $\heartsuit K$, $\spadesuit K$, $\clubsuit K$ at foundations
- Follow third subgoal: getting $\spadesuit K$ to foundations requires first having $\spadesuit Q$ at foundations and having $\spadesuit K$ movable.
- Follow second subsubgoal: having $\spadesuit K$ movable requires. . .

Critical Path Heuristics in the Literature

Critical path heuristics in the literature (admissible in red):

- h^m heuristic family (Haslum & Geffner, AIPS 2000)
- additive h^m (Haslum et al., AAAI 2005)
- additive-disjunctive heuristic graphs (Coles et al., ICAPS 2008)
- combination with delete relaxation (Fickert et al., JAIR 2016)

in Fast Downward

`hm(m=2)`, `hm(m=3)`, ...

warning: very inefficient implementation!