580 2011-09-13

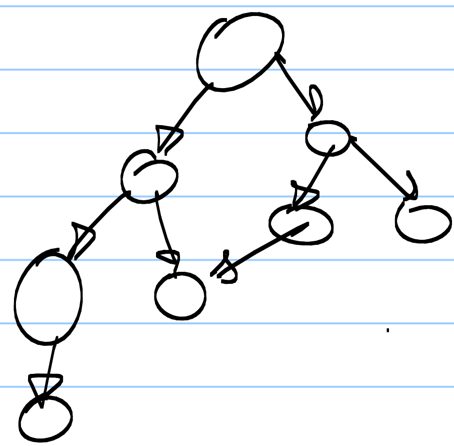
Note Title 2011-09-13

Graph search by dynamic programming

Let cost_to_goal(n) be the actual minimum cost to the goal node from node n $(\hat{h}(n))$.

$$\text{cost\_to\_goal}(n) = \begin{cases} \text{cost\_to\_goal}(n) = 0 & \text{if } n \text{ is goal} \\ \min_{\langle n,m\rangle \in A} (\text{cost}(n,m) + \text{cost\_to\_goal}(m)) & \text{otherwise} \end{cases}$$

Poole uses "lowest-cost search" for Dijkstra's algorithm, which is also called the uniform cost method.



Where do heuristics come from?

A: from solving relaxed (sub) problems!

A relaxed problem has the same nodes of the original one but it has more edges. (Informal, "syntactic" definition)

If you describe a state-space search problem by specifying constraints on the legal moves between states, a relaxed problem is one with fewer constraints.

Somerville et al., 1975 ~

$P = (\underline{A}, \underline{V}, \Pi, \Delta, \underline{s}, \underline{k})$

$\underline{A}$: set of attributes, $\underline{V}$ = set of values,

Π = set of constraints on states

Δ = set of constraints on moves,

$\underline{s}$ = start state

$\underline{k}$ = goal state

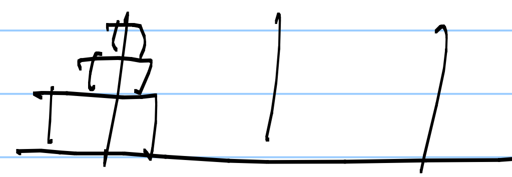
$\underline{s}$ and $\underline{k}$ are underlined, b/c they have a structure: they are lists of attribute-value pairs

Example: Towers of Hanoi ($\sim$1975)

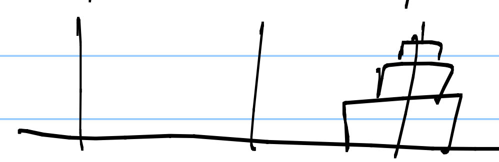
$\underline{A} = \{d_1, d_2, d_3\}$ (the disks $d_1 > d_2 > d_3$)

$\underline{V} = \{p_1, p_2, p_3\} \times \{h_1, h_2, h_3\}$ (pegs and heights)

$\underline{s} = ((d_1, (p_1, h_1)), (d_2, (p_1, h_2)), (d_3, (p_1, h_3)))$



$\underline{k} = ((d_1, (p_3, h_1)), (d_2, (p_3, h_2)), (d_3, (p_3, h_3)))$



$\Pi = \{ \pi_1, \pi_2 \}$

$\pi_1 =$ "every disk must be either on the table or over another disk" [no "floating" disks]

$\pi_2 =$ "if a disk is over another disk, the lower disk is bigger"

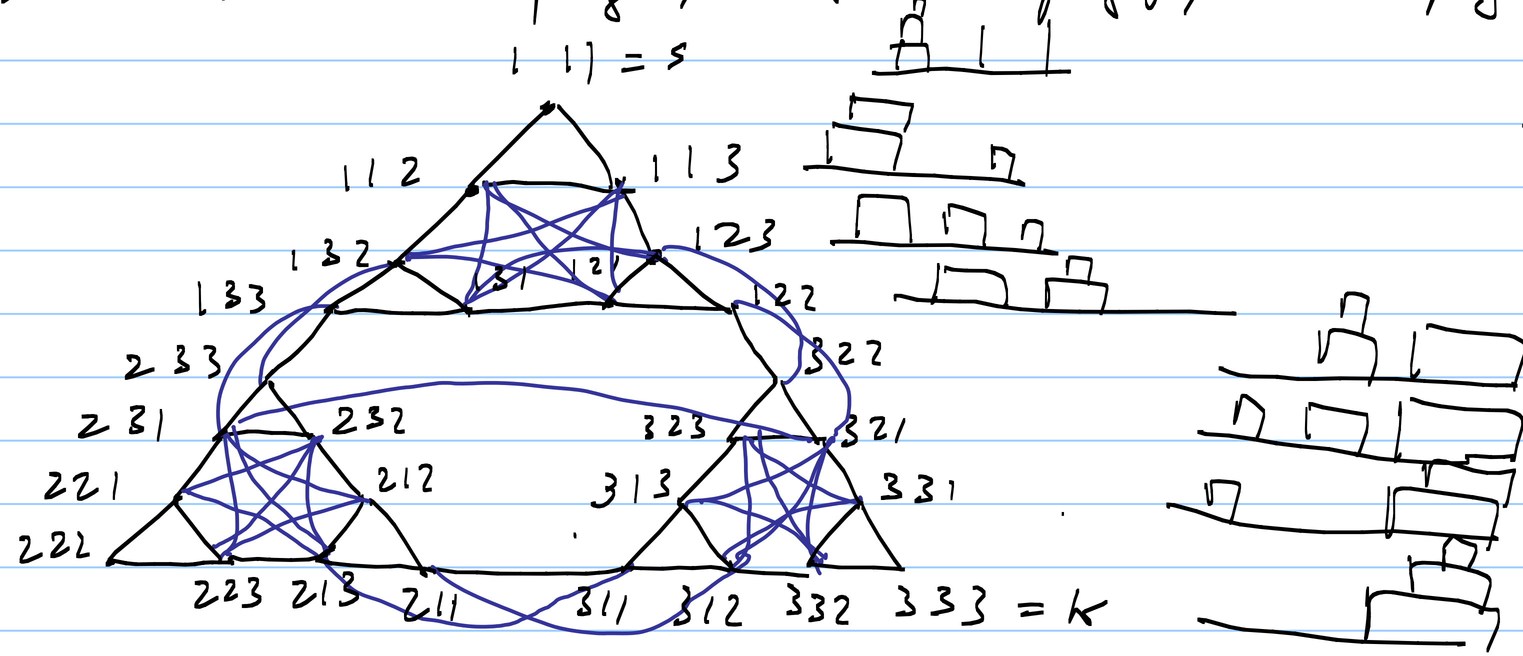
$\Lambda = \{ \lambda_1, \lambda_2, \lambda_3 \}$

$\lambda_1 =$ "only one disk can be moved at one time"

$\lambda_2 =$ "only the top disk on a peg can be moved"

$\lambda_3 =$ "a disk must be placed over the table or over a disk that was there before the move". $1 \le i, j, k \le 3$

Here a representation of the state space, where ijk means: disk 1 is on peg i, disk 2 on peg j, disk 3 on peg k



For a relaxed problem, we remove ∂_1.
The new moves are in indigo.

How good is the heuristic derived from the relaxed problem? A: very good!

		g	h	f = g+h	exp by Dijkstra?	exp by A*
p^*	111	0	6	6	y	y
	113	1	5		y	y
	123	2	4		y	y
	122	3	4		y	y
	322	4	3		y	y
	321	5	2		y	y
	331	6	1		y	y
	333	7	0	7	y	y
	112	1	6	7	y	maybe
	132	2	5	7	y	maybe
	133	3	6	9	y	N
	131	3	5	8	y	N

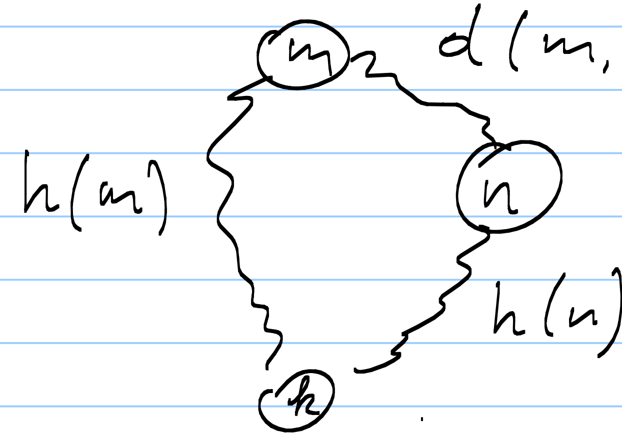
121	3	4	7	Y	maybe
233	4	4	8	Y	N
231	5	3	8	Y	N
232	5	4	9	Y	N
323	5	3	8	Y	N
221	6	4	10	Y	N
212	6	>1	>7	Y	N
313	6	>1	>7	Y	N
222	7	>1	>7	Y	N

Is this heuristic monotone (or consistent)?

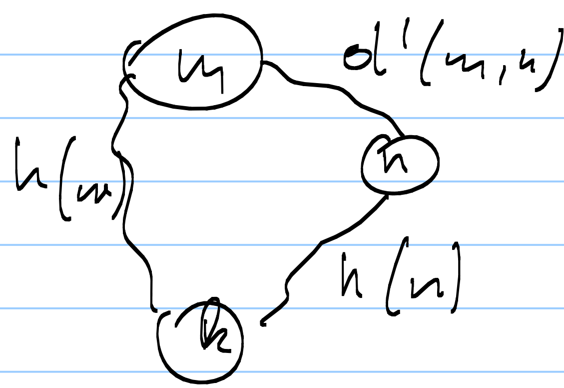
It is, $h(m) - h(n) \le d(m, n) \quad \forall n \in succs(m)$

General result: Every heuristic computed by problem relaxation is monotone.

Proof.



$d(m,n)$ is an upper bound $d'(m,n)$ on the distance from m to n in the relaxed problem used to compute h.



$$
\begin{aligned}
h(m) &\le h(n) + d'(m,n) \\
&\le h(n) + d(m,n)
\end{aligned}
$$

$\square$