Heuristic Hill-Climbing as a Markov Process

Carlos Linares López

Planning and Learning Group
Departamento de Informática
Avenida de la Universidad, 30
28911 - Leganés
Madrid (España)

carlos.linares@uc3m.es
http://www.plg.inf.uc3m.es/˜clinares

June, 12, 2008

Outline

Outline

Heuristic Hill-Climbing

```
HILL-CLIMBING (n:  NODE): COST
  if (n = t) return 0
  compute n_k = argmin{h(n_i)}, ∀n_i ∈ SCS(n)
  return k(n, n_k) + HILL-CLIMBING (n_k)
```

Figure: Pseudocode of the Heuristic Hill-Climbing search algorithm

- *Heuristic Hill-Climbing* commits to any of the descendants with the lowest heuristic estimation to t, underestimating all the other successors
- Heuristic Hill-Climbing exhibits the *Markovian property*

Heuristic Hill-Climbing

```
HILL-CLIMBING (n: NODE): COST
   if (n = t) return 0
   compute n_k = argmin{h(n_i)}, ∀n_i ∈ SCS(n)
   return k(n, n_k) + HILL-CLIMBING (n_k)
```

Figure: Pseudocode of the Heuristic Hill-Climbing search algorithm

- *Heuristic Hill-Climbing* commits to any of the descendants with the lowest heuristic estimation to t, underestimating all the other successors
- Heuristic Hill-Climbing exhibits the *Markovian property*

An example

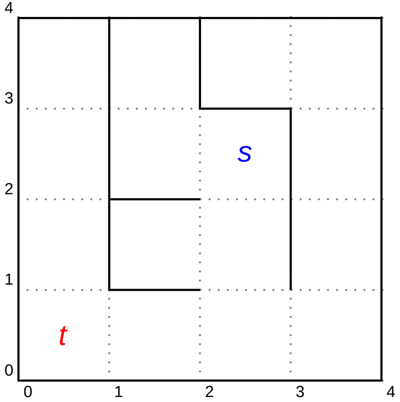

Figure: Searching in a maze 4×4

Outline

Homogeneous discrete-time Markov chains

- Let X_i denote the successive observations at time steps i, $i + 1$, etc.
- The Markovian property states that:

$$P(X_{n+1} = x_{n+1} | X_0 = x_0, X_1 = x_1, \ldots, X_n = x_n) = P(X_{n+1} = x_{n+1} | X_n = x_n)$$

- The matrix $P = \{p_{ij}\} = \{P(X_{n+1} = j | X_n = i)\}$ is a *stochastic matrix* that represents the probability of every single-step transition

Markovian states

- Two *nodes*, n and m, are mapped to the same *markovian state*, x, if they have the same chances to evolve to t
- The heuristic value, h, and branching factor, b, have been used in the past to discriminate nodes this way
- But also, the class c of a node n (i.e., the number of eligible descendants) shall be considered
- In fact, if $c(n) = 0$, **an error of the heuristic function has been found!**

Markovian states

- Two *nodes*, n and m, are mapped to the same *markovian state*, x, if they have the same chances to evolve to t
- The heuristic value, h, and branching factor, b, have been used in the past to discriminate nodes this way
- But also, the class c of a node n (i.e., the number of eligible descendants) shall be considered
- In fact, if $c(n) = 0$, **an error of the heuristic function has been found!**

An example

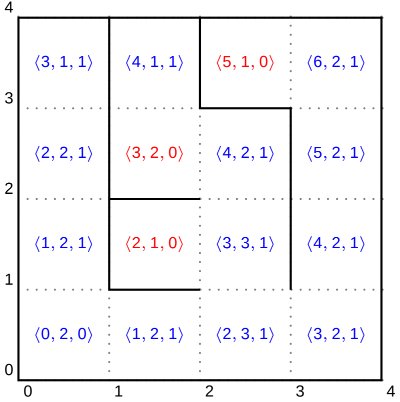

Figure: Markov states represented as $\tau\langle h, b, c\rangle$

Sampling the state space

- The probability to expand a given successor, $m \mapsto x_j$, of a node $n \mapsto x_i$ is:

$$q_{ij} = \begin{cases} \frac{1}{c(n)} & , c(n) > 0 \\ \frac{1}{b} & , c(n) = 0 \end{cases}$$

- Thus, the probability to step from markovian state x_i to x_j is:

$$p_{ij} = \frac{\sum_{i,j} q_{ij}}{\sum_{j} q_{ij}}$$

An example

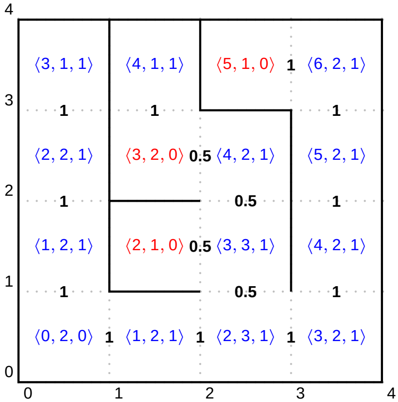

Figure: Computation of q_{ij}

An example: single-step transition matrix P

$$P = \begin{pmatrix}
\langle 0,2,0\rangle : & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\langle 1,2,1\rangle : & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\langle 2,1,0\rangle : & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\langle 2,2,1\rangle : & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\langle 2,3,1\rangle : & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\langle 3,1,1\rangle : & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\langle 3,2,0\rangle : & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 \\
\langle 3,2,1\rangle : & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\langle 3,3,1\rangle : & 0 & 0 & 0.5 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\langle 4,1,1\rangle : & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\langle 4,2,1\rangle : & 0 & 0 & 0 & 0 & 0 & 0 & 0.25 & 0.5 & 0.25 & 0 & 0 & 0 & 0 & 0 \\
\langle 5,1,0\rangle : & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\langle 5,2,1\rangle : & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\langle 6,2,1\rangle : & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{pmatrix}$$

Outline

Accuracy of $h(\cdot)$

Chapman-Kolmogorov equations

$$p_{ij}^{(\lambda)} = \sum_{\forall k} p_{ik}^{(l)} p_{kj}^{(\lambda-l)}$$

is the probability that a node in markovian state τ_i can get to a node in markovian state τ_j in exactly λ steps

Thus, $p_{x0}^{(x)}$ *is the probability that $h(\cdot)$ provided perfectly informed values* ***all along the path to the target node*** *for nodes in markovian state τ_x*

Mean solution length

Mean first passage times

$$M_{ij} = \sum_{\lambda=1}^{\infty} \lambda f_{ij}^{(\lambda)}$$

where

$$f_{ij}^{(\lambda)} = p_{ij}^{(\lambda)} - \sum_{l=1}^{\lambda-1} f_{ij}^{(l)} p_{jj}^{(\lambda-l)}$$

is the probability that, starting from markovian state τ_i, the first arrival to state τ_j occurs in exactly λ steps

Thus, M_{i0} *is the mean number of steps to reach the goal state or,* ***the mean solution length***

Outline

Results

	3×3 (181440 nodes)		2×5 (1814400 nodes)		4×4 (10^{13} nodes)	
h	Obs. ($\bar{p_x}$)	Pred. ($\hat{p_x}$)	Obs. ($\bar{p_x}$)	Pred. ($\hat{p_x}$)	Obs. ($\bar{p_x}$)	Pred. ($\hat{p_x}$)
0	1	1	1	1	1	1
1	1	1	1	1	1	1
2	1	1	1	1	1	1
3	0.8	0.8	0.75	0.75	0.714286	0.714286
4	0.13913	0.13913	0.097345	0.097345	0.062015	0.062015
5	0.073170	0.068879	0.065934	0.062834	0.052	0.050238
6	0.041726	0.036011	0.036710	0.031671	0.026440	0.025341
7	0.030864	0.025866	0.021269	0.021315	0.016626	0.016534
8	0.015230	0.011262	0.008424	0.009189	0.006601	0.006764
9	0.010228	0.007179	0.005080	0.005606	0.004788	0.004858
10	0.006235	0.003555	0.002983	0.002978	0.004843	0.004806
11	0.004706	0.002504	0.001833	0.001826		
12	0.002991	0.001266	0.001044	0.001026		
13	0.002023	0.000873	0.000630	0.000624		
14	0.001432	0.000470	0.000381	0.000371		
15	0.001081	0.000319	0.000217	0.000223		
16	0.000912	0.000178	0.000123	0.000134		
17	0.000765	0.000119	6.5×10^{-5}	8.0×10^{-5}		
18	0.000821	7.0×10^{-5}	3.2×10^{-5}	4.8×10^{-5}		
19	0.000790	4.4×10^{-5}	1.5×10^{-5}	2.8×10^{-5}		

Table: Results in different sliding tile puzzles

Bibliography I

Ghallab, M., Nau, D., Traverso, P.:
Automated Planning: Theory and Practice.
Morgan Kaufmann, San Francisco, CA, USA (2004)

Bonet, B., Geffner, H.:
Planning as heuristic search.
Artificial Intelligence **129**(1–2) (2001) 5–33

Bonet, B., Geffner, H.:
Planning as Heuristic Search: New Results.
In: Recent Advances in AI Planning: Proceedings of the Fifth European Conference on Planning (ECP-99). Volume Lecture Notes in AI 1809. Springer (1999) 359–371

Bibliography II

Hoffmann, J., Nebel, B.:
The FF planning system: Fast plan generation through heuristic search.
Journal of Artificial Intelligence Research **14** (2001) 253–302

Refanidis, I., Vlahavas, I.:
The grt planning system: Backward heuristic construction in forward state-space planning.
Journal of the Artificial Intelligence Research **15** (2001) 115–161

Bibliography III

Korf, R.E.:
Search.
In: Encyclopedia of Artificial Intelligence. Volume 2. John Wiley & Sons (1987) 994–998

Stewart, W.J.:
Introduction to the Numerical Solution of Markov Chains.
Princeton University Press, Princeton, New Jersey, USA (1994)

Korf, R.E., Reid, M., Edelkamp, S.:
Time complexity of iterative-deepening-A*.
Artificial Intelligence **129**(1–2) (2001) 199–218