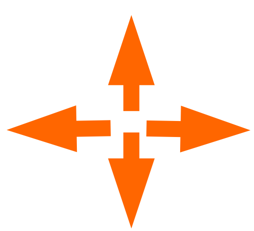
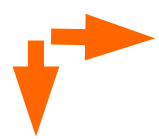
Funzioni euristiche

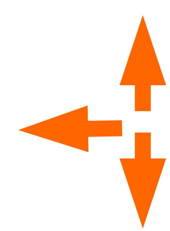
- Studiamo la natura delle euristiche usando il gioco dell’8, uno dei primi problemi sui quali si è sperimentata la ricerca informata
- In media, generando in modo casuale lo stato iniziale:
  - occorrono 22 mosse per arrivare alla soluzione
  - Il branching factor è pari a 3:



Ho 4 mosse per la tessera centrale



Ho 2 mosse per gli spigoli

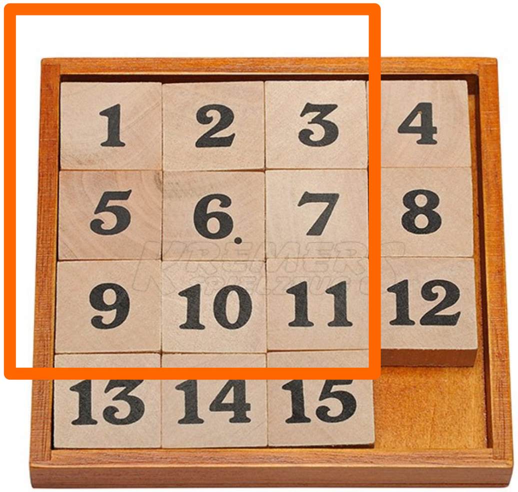


Ho 3 mosse per le tessere sui lati

Albero e grafo esaustivo di ricerca

- Albero esaustivo di ricerca:
  contiene 3^{22} nodi (oltre 30.000.000.000)
- Grafo esaustivo di ricerca:
  contiene “solo”~ 180.000 nodi, perché si evitano i duplicati
- E se passiamo dal problema dell’8 al problema del 15? Non sembra molto più complesso, invece:
  - Il grafo esplode: avrebbe circa 10^{23} nodi

Problema dell’8



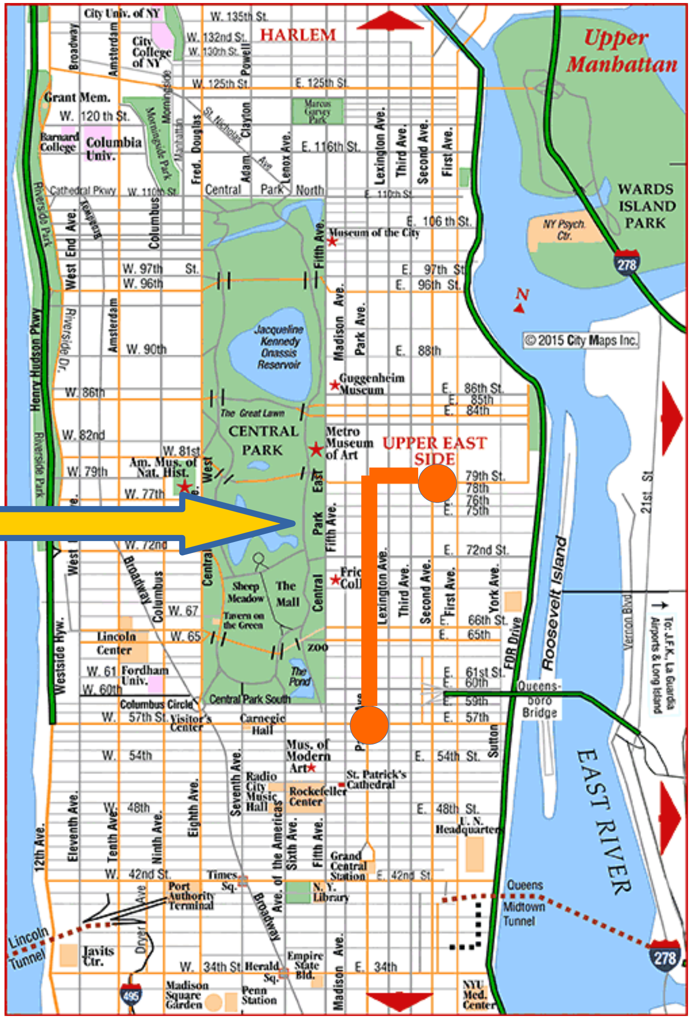
Problema del 15

Slocum and Sonnenveld, The 15 Puzzle Book: How it drove the world crazy, 2006

http://www.slocumpuzzles.com/index2.html

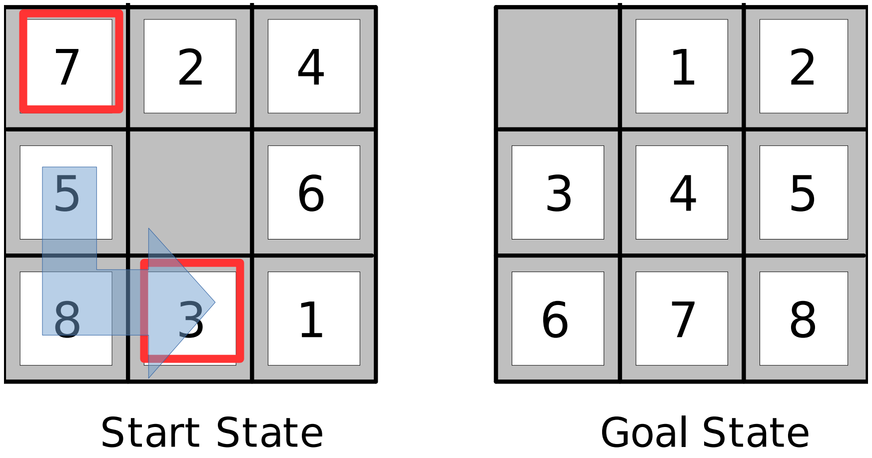
A*: euristiche per il problema del 15 (dell'8)

- A* necessita di euristiche ammissibili, cioè tali da non sovrastimare mai il numero dei passi che portano al goal
- Due possibili euristiche:
  - **h1 = numero di tessere fuori posto.** È ammissibile perché ogni tessera fuori posto deve essere spostata almeno una volta.
  - **h2 = distanza di Manhattan (o block distance).** È la somma della distanza di una tessera dalla sua posizione desiderata, contata in numero di tessere attraversate (originariamente di isolati attraversati) sulle ascisse più numero di tessere attraversate sulle ordinate. È ammissibile perché ogni mossa può spostare una tessera al più di una posizione più vicina al goal.



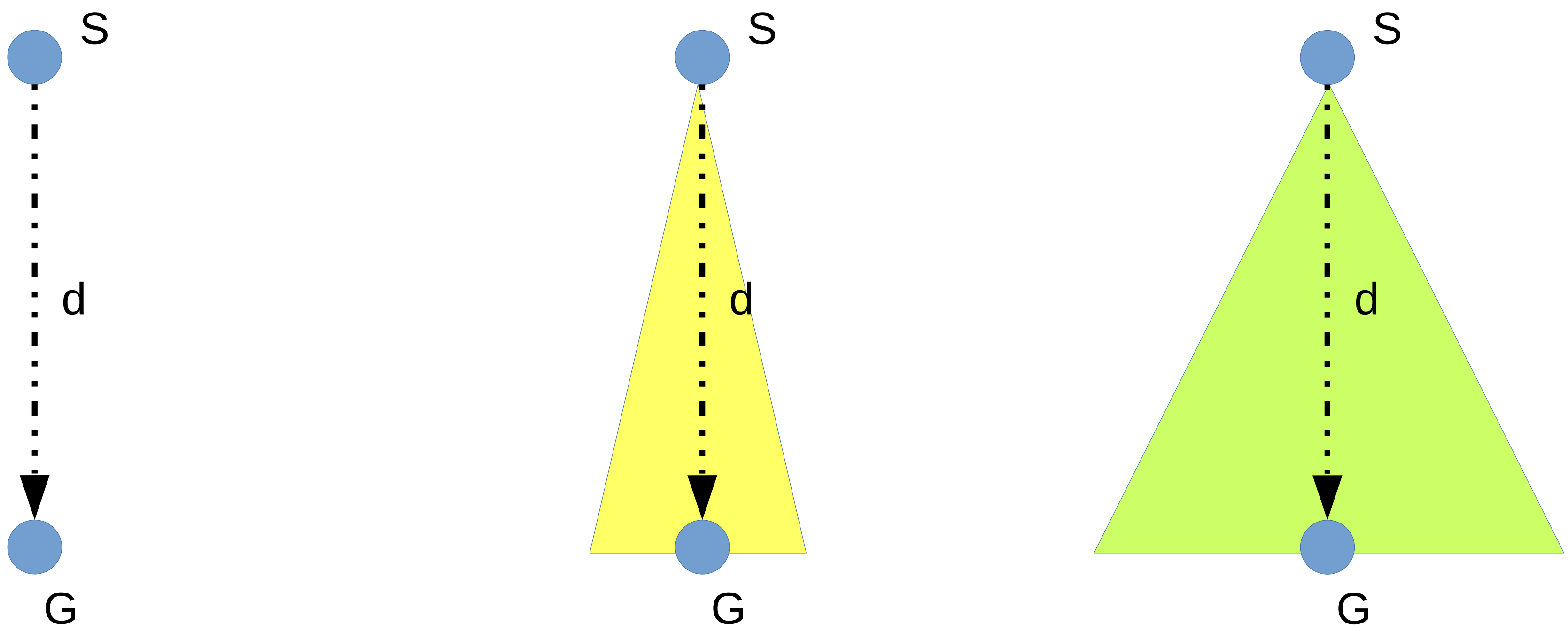
Esempio

- h1(s) = 8
  tutte le tessere sono fuori posto
- h2(s) = 3 + 1 + 2 + ... = 18
  si sommano le distanze di Manhattan calcolate per ogni tessera



Confronto sperimentale

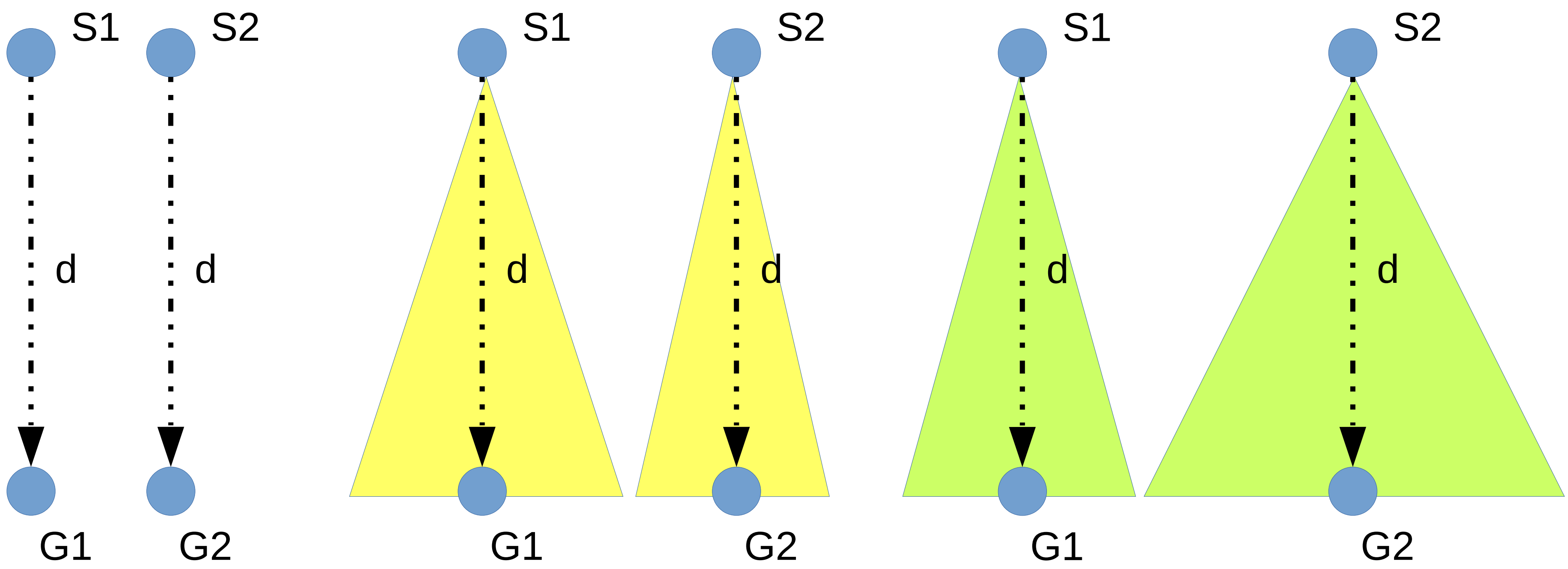
Scopo: vogliamo decidere quale sia l'euristica migliore basandoci sui risultati derivanti dal loro utilizzo, in particolare i numeri di nodi che uno stesso algoritmo di ricerca informato (nel nostro caso A*) produce



Sia d la profondità della soluzione

Euristiche diverse causeranno in generale l'esplorazione di numeri di nodi differenti.

Ogni istanza produce un risultato: come combinare questi dati?



Se considero due diverse istanze del problema con soluzioni a pari profondità

Il numero di nodi espansi su ciascuna istanza del problema cambia perché anche stato iniziale e goal hanno un'influenza. Come confrontare le euristiche in presenza di tanta variabilità?

Valutazione sperimentale

- La valutazione sperimentale delle euristiche h comprende i seguenti passi:
  - Generare un numero significativo di casi
  - Applicare lo stesso algoritmo di ricerca a ogni caso, tante volte quante sono le euristiche da valutare (una per ogni euristica)
  - Raccogliere i dati risultanti (numero di nodi generati, profondità della soluzione …)
  - Calcolare i valori medi dei risultati ottenuti in casi affini (esempio quelli in cui la profondità della soluzione è la stessa)
  - Valutare e confrontare le prestazioni

Qualità delle euristiche

- La qualità di un’euristica può essere calcolata computando il **branching factor effettivo b***
- Supponiamo di avere eseguito A* su un certo problema, siano:
  - N = numero di nodi generati a partire da un nodo iniziale
  - d = profondità della soluzione trovata
- **b* = branching factor di un albero uniforme di profondità d che contiene N+1 nodi**
- $N+1 = 1 + (b^*) + (b^*)^2 + \ldots + (b^*)^d$

Branching factor effettivo (una stima)

$$N + 1 = 1 + b^* + (b^*)^2 + \ldots + (b^*)^d$$

$$N + 1 = ((b^*)^{d+1} - 1)/(b^* - 1)$$

$$N \approx (b^*)^d \Rightarrow b^* \approx \sqrt[d]{N}$$

Valutazione sperimentale: qualità delle euristiche

- Qualità di un’euristica calcolata a posteriori, a partire da alcuni casi d’uso, cioè problemi in cui viene applicato A*
- Ogni istanza può produrre b* differenti ma tali valori saranno tendenzialmente consistenti
- Quindi bastano alcune misure su un campione (piccolo insieme di problemi) per calcolare la bontà di un’euristica
- Le **euristiche migliori** hanno b* bassi, vicini a 1
- Esse permettono di risolvere problemi complessi in tempi ragionevoli

Esempio: meglio h1 o h2?

- Sono stati generati in modo casuale 1200 problemi del 15 con profondità di soluzione compresa fra 2 e 24
- Sono stati risolti con Iterative Deepening e A*, usando h1 e poi h2
- I numeri di nodi generati e l’effective branching factor sono stati calcolati caso per caso
- Sono state prodotte le medie per ogni profondità di soluzione

Esempio

	Search Cost (nodes generated)			Effective Branching Factor		
d	IDS	$A^*(h_1)$	$A^*(h_2)$	IDS	$A^*(h_1)$	$A^*(h_2)$
2	10	6	6	2.45	1.79	1.79
4	112	13	12	2.87	1.48	1.45
6	680	20	18	2.73	1.34	1.30
8	6384	39	25	2.80	1.33	1.24
10	47127	93	39	2.79	1.38	1.22
12	3644035	227	73	2.78	1.42	1.24
14	–	539	113	–	1.44	1.23
16	–	1301	211	–	1.45	1.25
18	–	3056	363	–	1.46	1.26
20	–	7276	676	–	1.47	1.27
22	–	18094	1219	–	1.48	1.28
24	–	39135	1641	–	1.48	1.26

Figure 3.29 Comparison of the search costs and effective branching factors for the ITERATIVE-DEEPENING-SEARCH and A* algorithms with h_1, h_2. Data are averaged over 100 instances of the 8-puzzle for each of various solution lengths d.

Esempio

	Search Cost (nodes generated)			Effective Branching Factor		
d	IDS	$A^*(h_1)$	$A^*(h_2)$	IDS	$A^*(h_1)$	$A^*(h_2)$
2	10	6	6	2.45	1.79	1.79
4	112	13	12	2.87	1.48	1.45
6	680	20	18	2.73	1.34	1.30
8	6384	39	25	2.80	1.33	1.24
10	47127	93	39	2.79	1.38	1.22
12	3644035	227	73	2.78	1.42	1.24
14	–	539	113	–	1.44	1.23
16	–	1301	211	–	1.45	1.25
18	–	3056	363	–	1.46	1.26
20	–	7276	676	–	1.47	1.27
22	–	18094	1219	–	1.48	1.28
24	–	39135	1641	–	1.48	1.26

Figure 3.29 Comparison of the search costs and effective branching factors for the ITERATIVE-DEEPENING-SEARCH and A* algorithms with h_1, h_2. Data are averaged over 100 instances of the 8-puzzle for each of various solution lengths d.

Nodi prodotti con A(h1) ≫ **Nodi prodotti con A(h2)**

b* per A(h1) ≫ **b* per A(h2)**

Valutazione teorica

- Decidere la qualità di un’euristica a posteriori richiede un lavoro sperimentale
- È possibile adottare un approccio differente, che consenta la valutazione a priori?
- Nell’esempio precedente h1 e h2 sono ammissibili, cioè per ogni n, $h^*(n) \geq h1(n)$ e $h^*(n) \geq h2(n)$
- Inoltre $h2(n) \geq h1(n)$ per definizione
- Queste proprietà sono utili per la decisione?

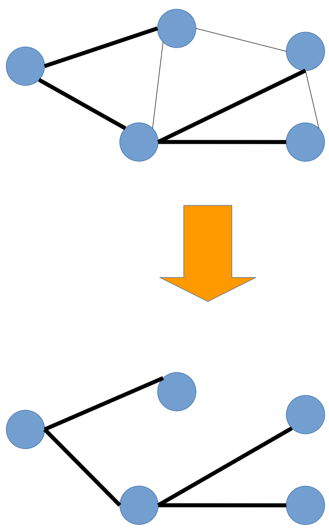
Euristiche dominanti

- Siano h1 e h2 due euristiche ammissibili tali che per ogni n h2(n) > h1(n). i dice che <u>h2 domina h1</u> o anche che <u>h2 è più informata di h1</u> perché approssima meglio h*
- Un'euristica più informata permette ad A* di giungere alla soluzione in modo più efficiente rispetto all'euristica meno informata
- Sia C* il costo della soluzione ottima, A* espande tutti i nodi con valore f(n) < C* (è un teorema che non dimostriamo), di qui, ricordando che f(n) = g(n)+h(n), si deriva che:
  - Tutti i nodi con h(n) < C* - g(n) saranno espansi
  - C* non dipende dall'euristica ma solo dal costo delle azioni, è costante
  - Poiché h2(n) > h1(n), usando h1, A* espanderà di sicuro almeno tutti i nodi che espande usando h2

Nota: sul libro la dominanza è data col ≥ ma strettamente parlando la maggiore efficienza si ottiene solo se h2>h1

Costruzione di euristiche ammissibili

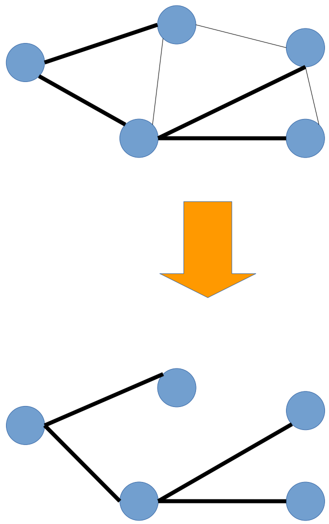
- Problemi e problemi rilassati: un problema ne rilassa un altro quando toglie qualche vincolo
- Il grafo degli stati di un problema rilassato è un supergrafo di quello del problema originario perché include transizioni che i vincoli di quest'ultimo non consentono (meno vincoli, più transizioni possibili)



Supergrafo del problema rilassato

Grafo del problema

Costruzione di euristiche ammissibili



Supergrafo del problema rilassato

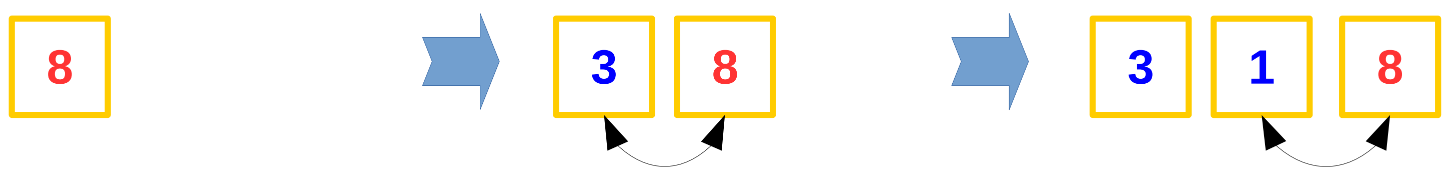
Grafo del problema

Poiché il supergrafo contiene il grafo del problema ed altri archi, contiene anche le soluzioni ottime al problema di nostro interesse

Il costo di una soluzione ottima al problema rilassato risulta essere un’euristica ammissibile per il problema di interesse

Esempio: gioco dell'8

- Una possibile versione rilassata consente di spostare le tessere anche in posizioni occupate, scambiando tessere



Rilassamenti automatici del problema dell'8

- Definiti i vincoli di un problema, è possibile costruire delle versioni <u>rilassate automaticamente per astrazione</u>, esempio:
  - Vincoli del problema dell'8
    - Una tessera può spostarsi da A a B se (1) A e B sono adiacenti e (2) se B è vuota
  - Tre possibili rilassamenti del problema (rimozione di vincoli):
    1) Rimozione del vincolo (2):
       una tessera può spostarsi da A a B se A è adiacente a B
    2) Rimozione del vincolo (1):
       una tessera può spostarsi da A a B se B è vuota
    3) Rimozione di tutti i vincoli:
       una tessera può spostarsi da A a B

ABSOLVER, 1993

- Absolver II(*) è un esempio di programma che è in grado di generare automaticamente euristiche ammissibili per astrazione:
  - Ha scoperto la prima euristica ammissibile per il cubo di Rubik
  - Ha scoperto un’euristica ammissibile per il problema dell’8 che è migliore di quelle precedentemente proposte
- Gli studi sulla generazione di euristiche continua oggi soprattutto nell’area di planning (costruzione automatica di piani)

(*) Machine Discovery of Effective Admissible Heuristics, Armand E. Prieditis

Euristiche non dominanti

- Quando si riesce a identificare un insieme di euristiche ammissibili, <u>nessuna delle quali risulta dominante</u> sulle altre, è possibile costruirne una dominante in questo modo:
  $h(n) = \max\{h1(n), h2(n), \ldots, hk(n)\}$
- l'euristica composta è <u>ammissibile</u> perché lo sono le euristiche che compone, è <u>dominante</u> su quelle composte per definizione
- Per ogni nodo viene restituita la valutazione più accurata

Alternativa: indurre euristiche

- Un approccio alternativo consiste nell’usare uno strumento per l’apprendimento automatico
- Molti di questi strumenti hanno un fondamento statistico e sono in grado di costruire della conoscenza da un insieme di esempi rappresentati da opportune feature (caratteristiche)
- Ogni stato viene arricchito delle feature scelte (es. numero di celle fuori posto, numero di vicini errati, …)
- Vengono generati casualmente dei problemi, che vengono risolti collezionando i dati (stati e loro caratterizzazioni + costo per raggiungere la soluzione)
- Viene applicato agli esempi un metodo di apprendimento per induzione che estrae un’euristica