Intelligenza Artificiale

Ricerca euristica
L'algoritmo A*

Marco Piastra

Ricerca non informata

- **Ricerca nello spazio degli stati**
  - Definizione di un grafo come spazio degli stati
    - I nodi rappresentano gli stati
    - Gli archi (direzionati) rappresentano le transizioni
    - Gli archi hanno un costo (per difetto = 1)
  - Ricerca come costruzione di un albero
    - Da uno stato di partenza ad uno stato goal
    - Dallo stato di partenza s si derivano tutti i Successors(s)
    - Si agisce iterativamente
      (secondo metodi diversi, p.es. breadth-first, depth-first)
    - Fino al raggiungimento dello stato goal
  - Generazione sistematica
    - I metodi di ricerca prevedono la generazione progressiva e sistematica degli stati fino a raggiungere lo stato goal
    - Non viene usata altra informazione
    - Spesso poco efficienti

Ricerca euristica

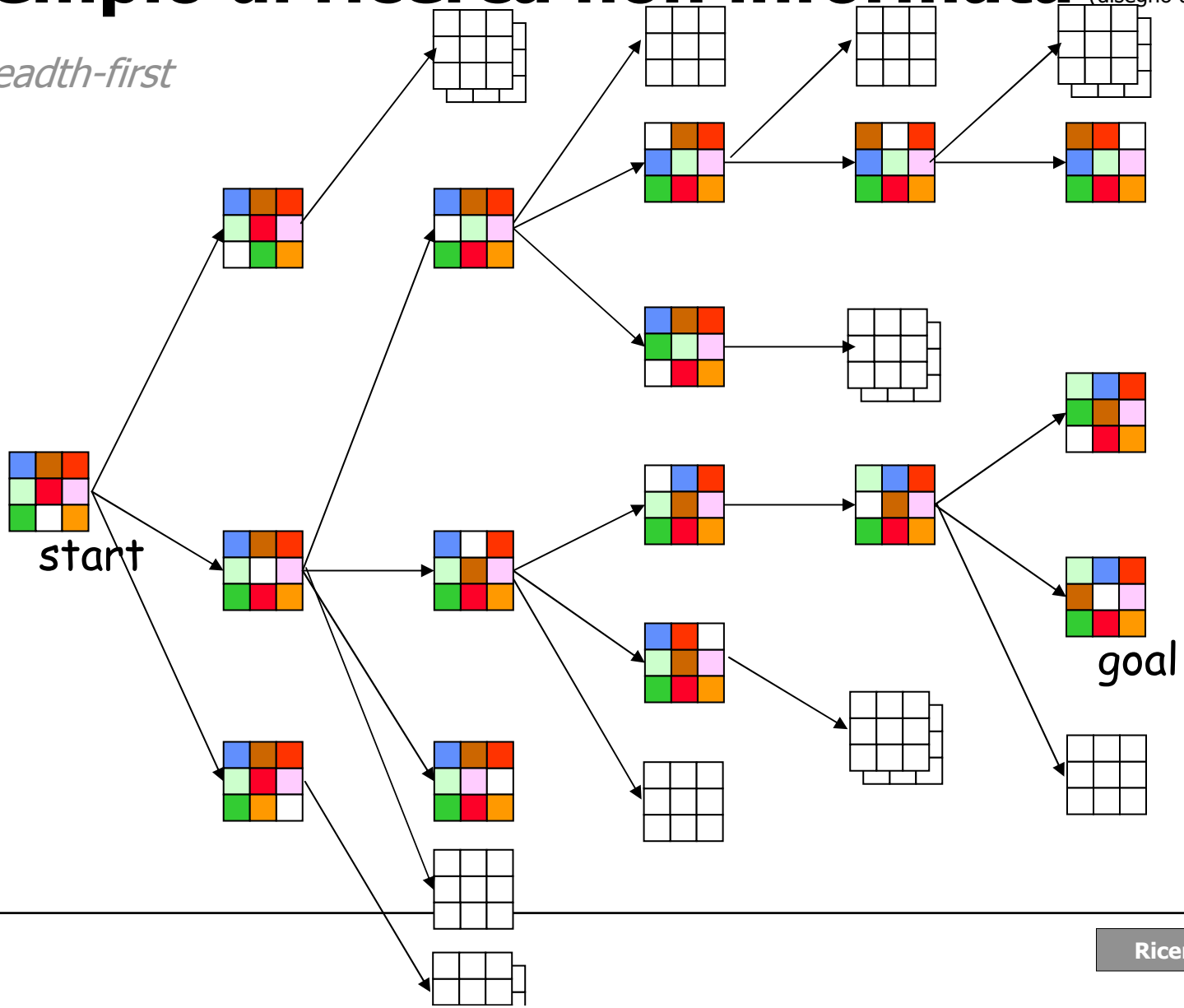
- **In greco, il verbo 'heuriskein' significa 'trovare'**

- **Uso dell'informazione disponibile**
  - Regole aggiuntive per migliorare il processo di ricerca
    - Dove 'migliorare' significa migliorare la performance **media**
  - Tipicamente, i metodi di ricerca euristica:
    - Utilizzano un metodo di ricerca di base, o non informata
    - Introducono regole aggiuntive (p.es. di preferenza)
    - Hanno la stessa complessità *worst-case* del metodo di base
  - Nei metodi di ricerca
    - Si utilizza una **funzione euristica** H(s):
      stima della distanza tra lo stato s e lo stato goal
    - La ricerca è guidata da una funzione di valutazione f(s)
      $f(s) = g(s) + h(s)$
    - g(s) è il costo del raggiungimento di s dallo stato iniziale
    - Il metodo è il *best-first:* come il metodo di ricerca a costo uniforme ma usando f(s) al posto di g(s)

Esempio di ricerca non informata

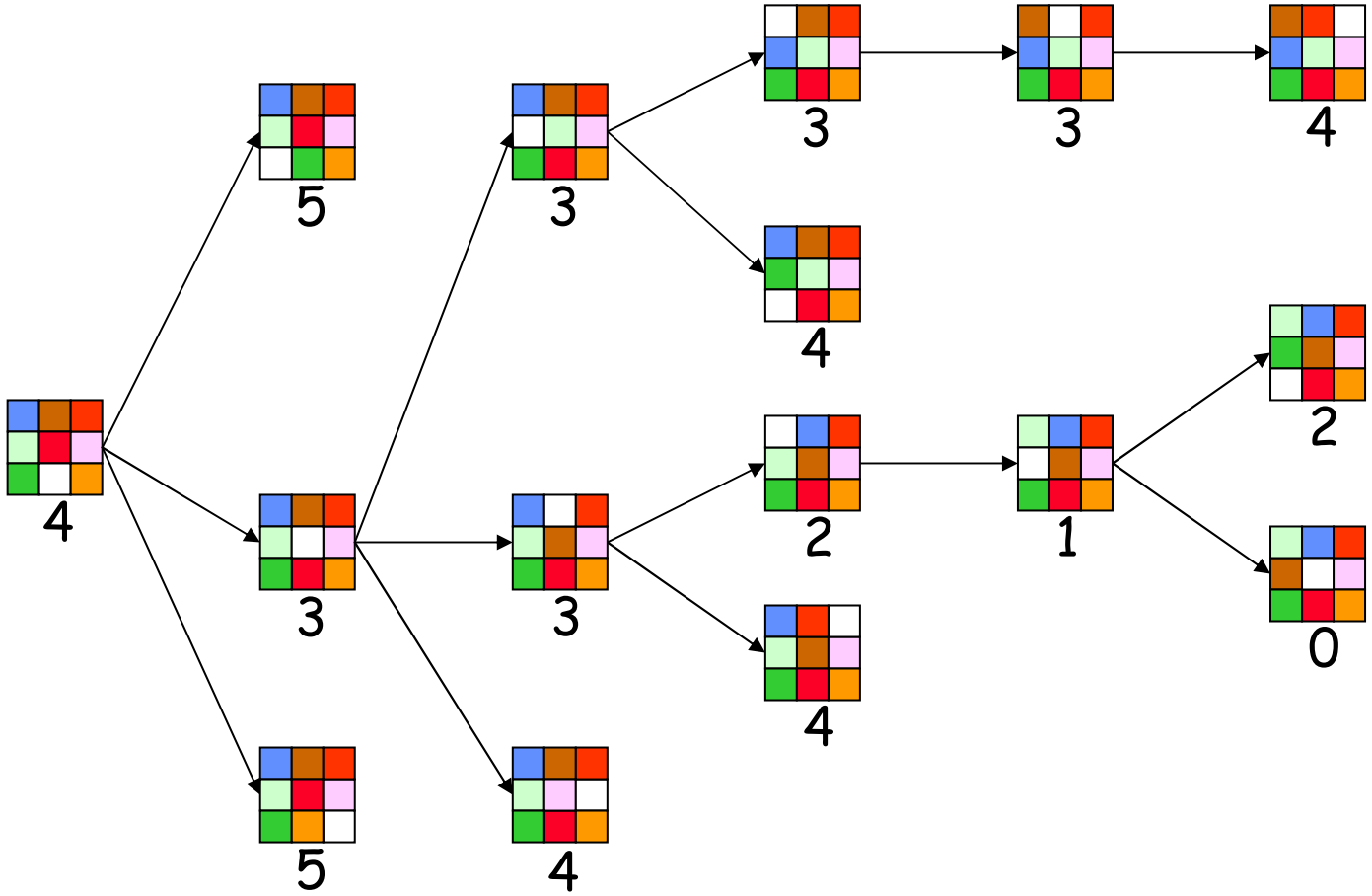
(disegno di J.C. Latombe)

breadth-first



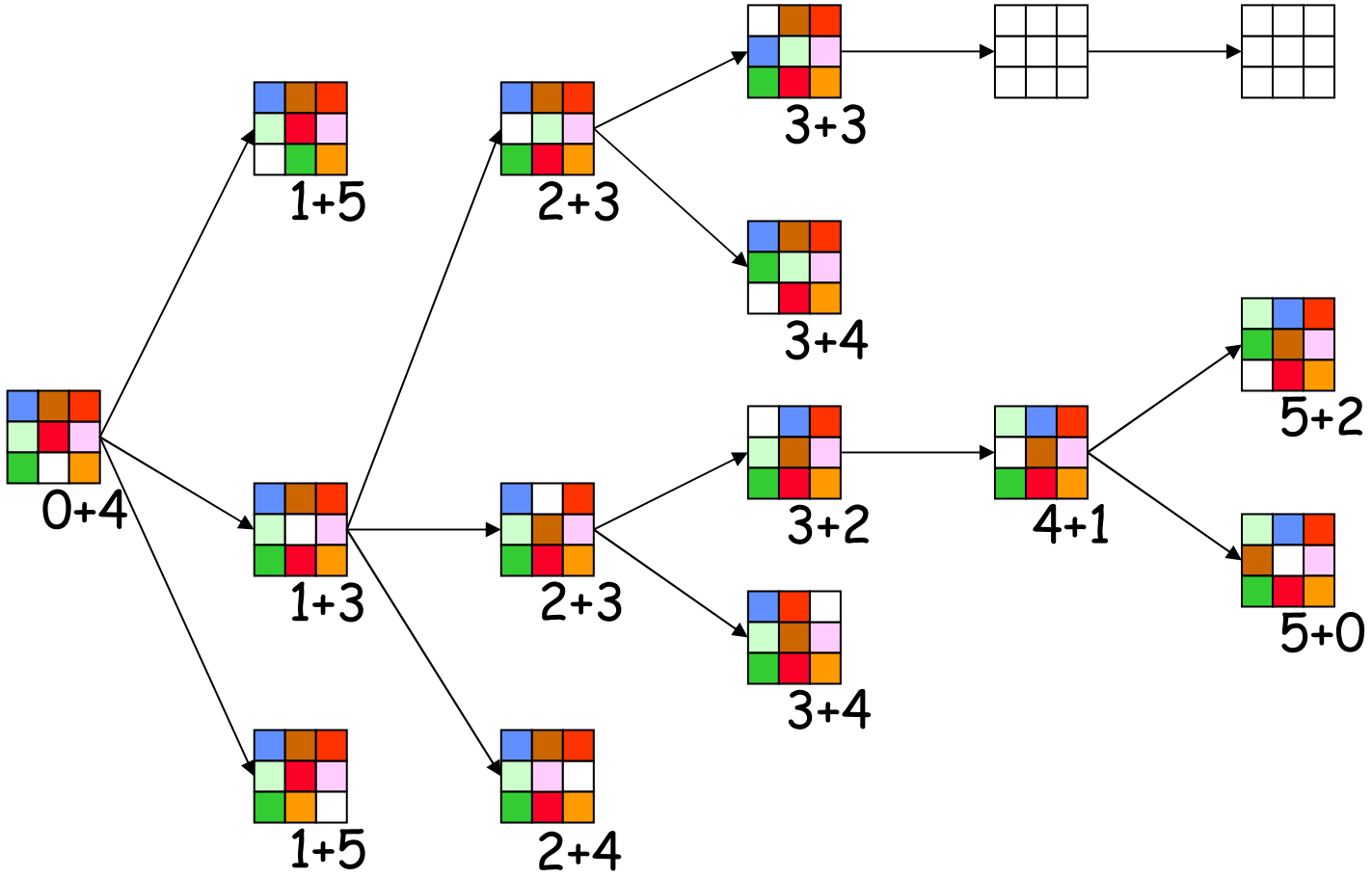
Esempio di ricerca euristica (1) (disegno di J.C. Latombe)

f(s) = h(s) numero di parti fuori posto



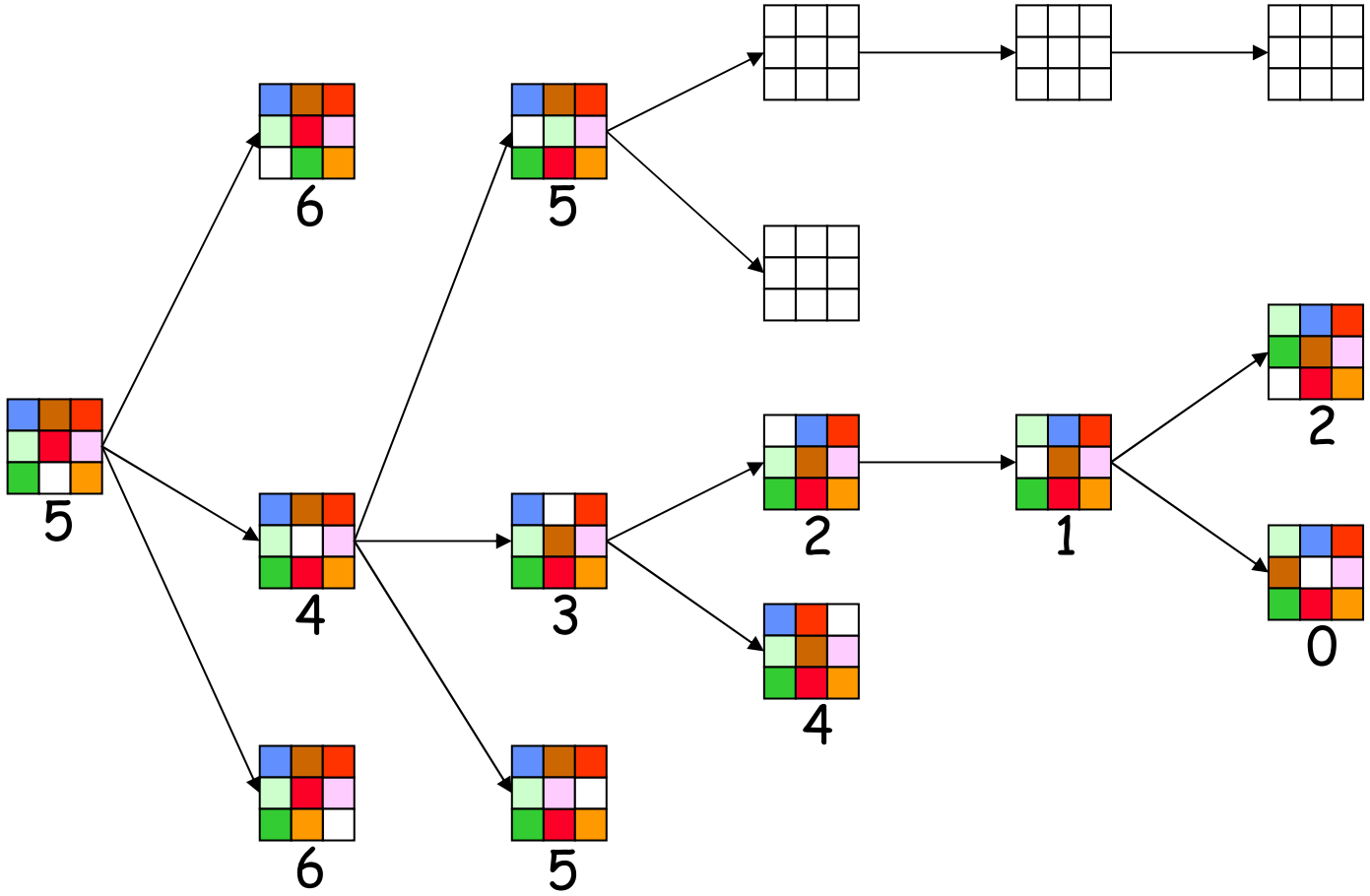
Esempio di ricerca euristica (2) (disegno di J.C. Latombe)

f(s) = g(s) + h(s) numero di parti fuori posto



Esempio di ricerca euristica (3) (disegno di J.C. Latombe)

f(s) = h(s) Σ distanza della singola parte dalla posizione goal



Algoritmo A* (pronuncia: 'a-star')

- Euristica **ammissibile**
  - Una funzione euristica h(s) è detta ammissibile sse $0 \leq h(s) \leq d(s)$
    - dove d(s) è la distanza minima effettiva tra s e lo stato goal
    - in altri termini, h(s) è ottimistica: stima d(s) per difetto
    - se s_g è lo stato goal, $h(s_g) = 0$
- Algoritmo A*
  - Il metodo di ricerca *best-first* che utilizza una f(s) = g(s) + h(s) dove h(s) è un'euristica ammissibile
  - L'algoritmo è **completo**
    - Se una soluzione esiste, l'algoritmo termina e la trova
  - L'algoritmo è **ottimale**
    - In presenza di soluzioni multiple, l'algoritmo è in grado di trovare la migliore
  - L'algoritmo è **efficiente**
    - Data un'euristica h, nessun altro algoritmo garantisce l'esplorazione di un minor numero di stati
  - Queste proprietà valgono sse la ricerca NON si ferma nei nodi già visitati

Euristica ammissibile

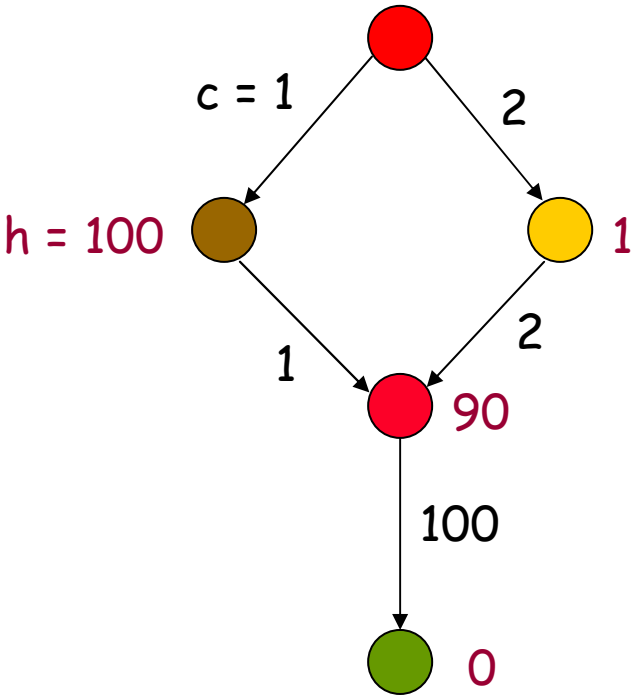
- In generale
  - Spesso coincide con la funzione costo di un problema con **meno** vincoli dell'originale
    - Tecnica di **rilassamento** dei vincoli
  - Esempio (gioco dell'otto):
    - Somma delle parti fuori posto
      si assume che ogni parte possa essere posizionata in una mossa
    - Sommatoria delle distanze delle singola parti dalla posizione goal
      si assume che il movimento di un parte non interferisca con il movimento delle altre (distanza a 'quadro vuoto')

Espansione ripetuta (1)

- Ricerca non informata
- Ricerca breadth-first
  - L'espansione ripetuta di stati già visitati può generare cicli
    - In generale, si utilizza una hashtable che memorizza gli stati già visitati
- Ricerca depth-first
  - Si espande un ramo alla volta
  - La verifica stati già visitati sul singolo ramo
    - Impedisce il generarsi di cicli
    - Non porta ad alcuna ottimizzazione
  - La verifica stati già visitati nel corso dell'intera ricerca
    - Diminuisce il numero medio di espansioni
    - Obbliga all'uso di una hashtable complessiva che, nel caso peggiore, ha un occupazione di memoria pari a quella necessaria per il breadth-first

Espansione ripetuta (2)

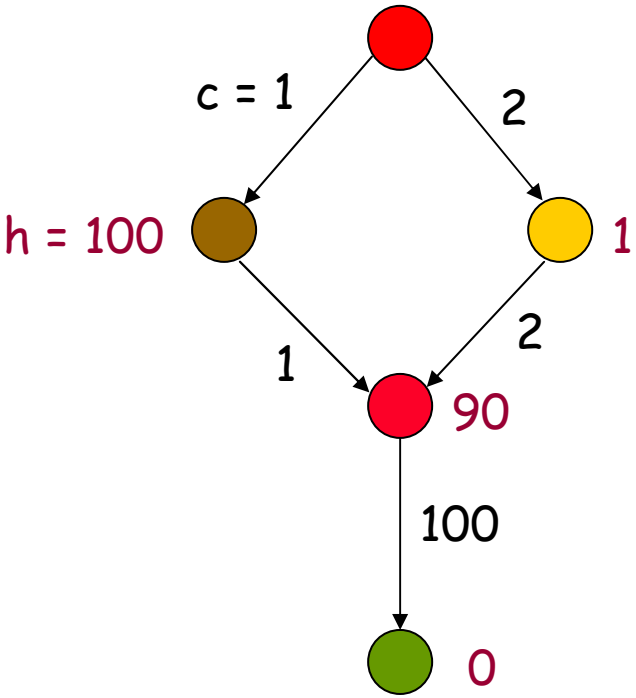
- Tutto cambia in una ricerca euristica …

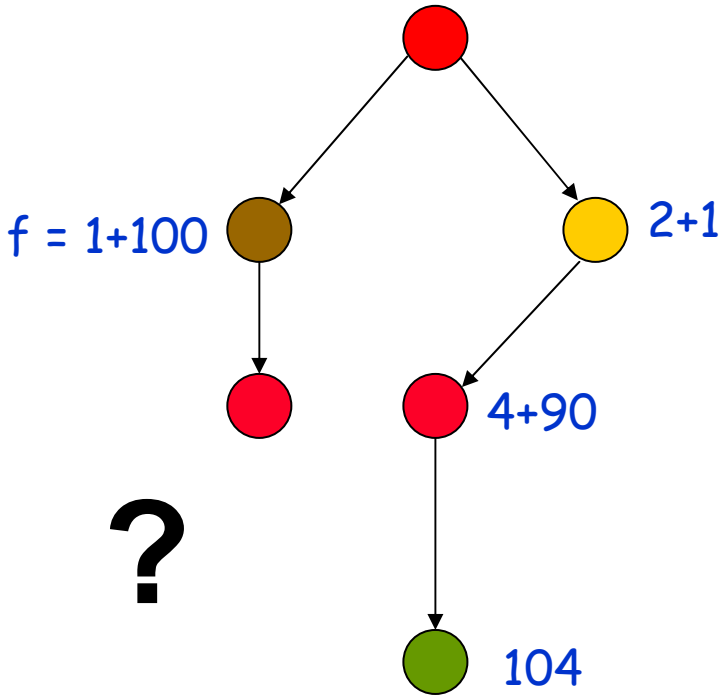


La funzione euristica h() è ammissibile

Espansione ripetuta (2)

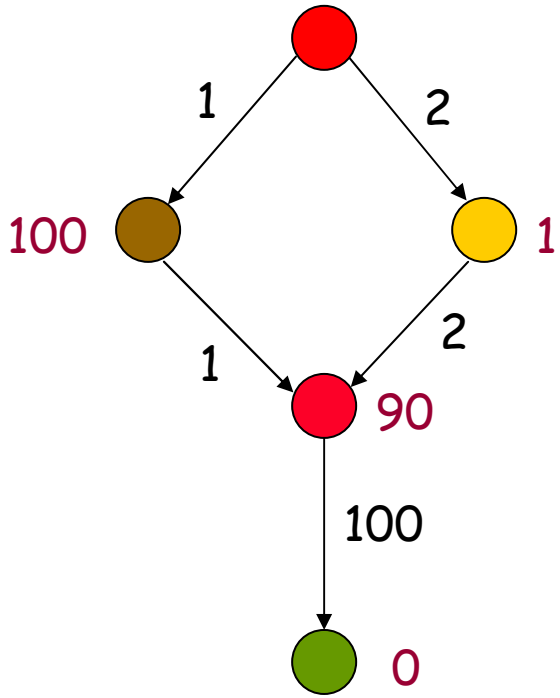
Se non si espande nuovamente il nodo già visitato, l'algoritmo procede e trova una soluzione non ottimale

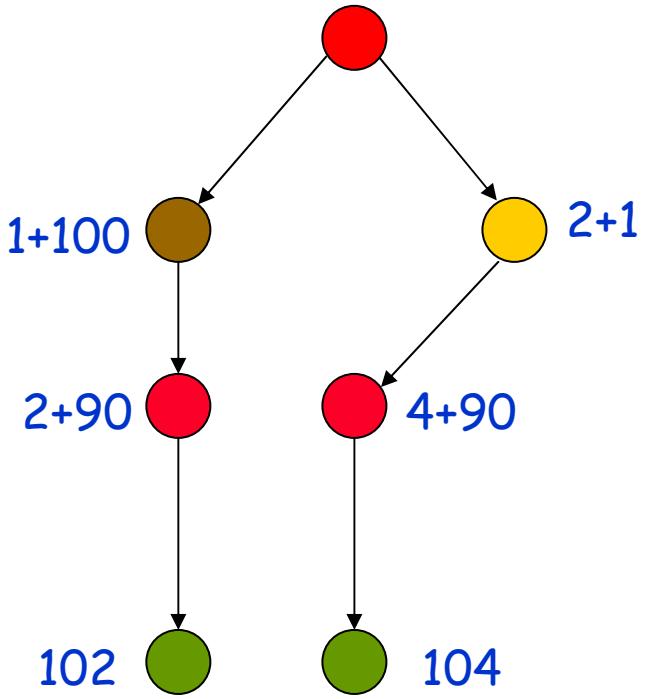




Espansione ripetuta (3)

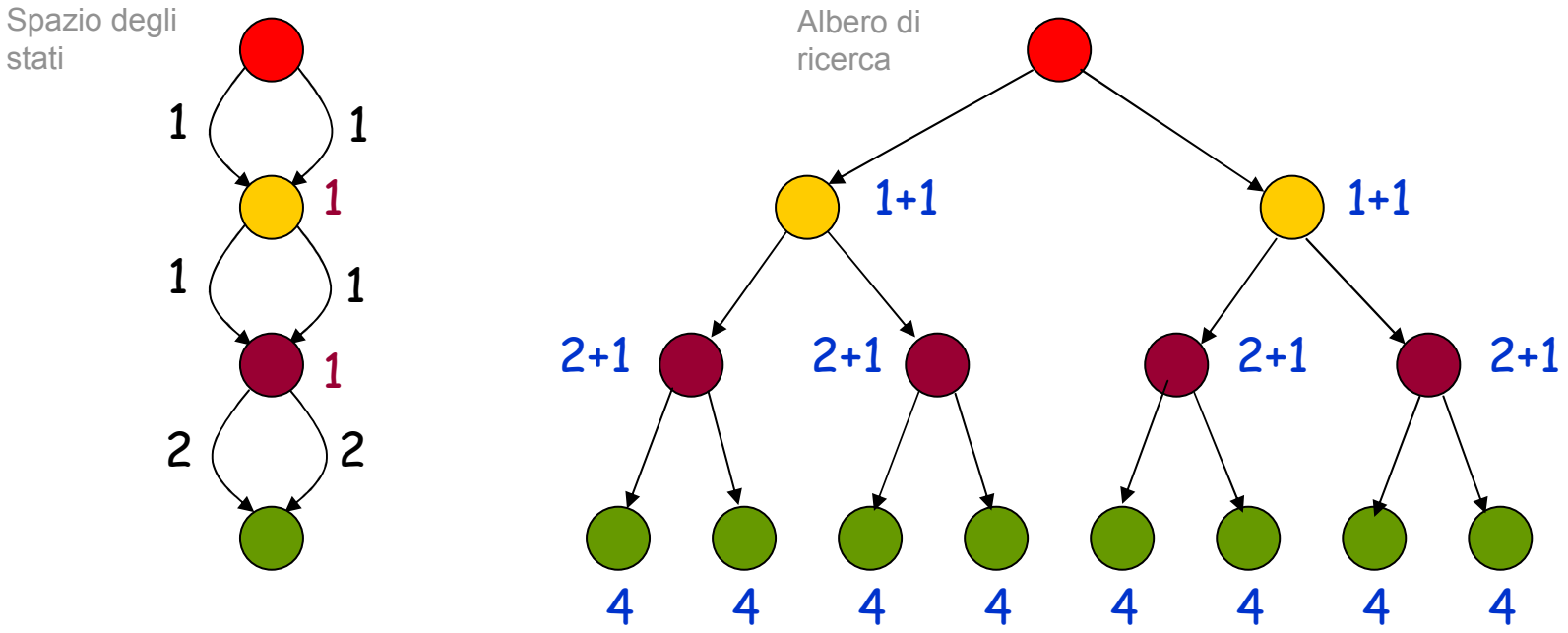
Viceversa, espandendo di nuovo anche i nodi già visitati, l'algoritmo trova la soluzione ottimale





Espansione ripetuta (4)

- Tuttavia …
  - L'espansione ripetuta provoca un aumento esponenziale delle dimensioni dell'albero di ricerca
    - (I cicli vengono identificati ed evitati comunque)



Espansione ripetuta: Algoritmo A*

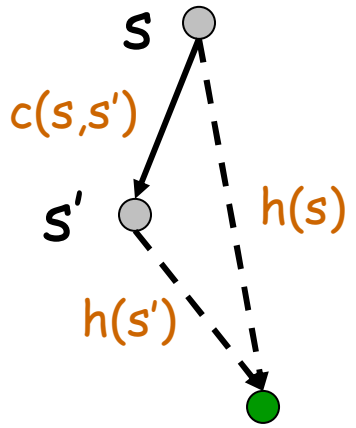
- Risultati formali
- In un metodo di ricerca euristica
  - L'espansione dei nodi già visitati può essere evitata senza conseguenze solo se il nuovo percorso verso lo stato ha un costo g superiore al costo dei percorsi già esplorati
- Algoritmo A*
  - Con questo accorgimento, l'Algoritmo A* rimane completo e ottimale
  - Ma la complessità di memoria può essere esponenziale

Funzione euristica consistente (1)

- Definizione
  - Una funzione euristica h(s) è **consistente** sse, per qualsiasi coppia s e s', s' ∈ Successors(s)

    $h(s) \leq c(s,s') + h(s')$

    dove c(s,s') è il costo associato alla transizione
  - h(s) si dice anche **monotona**

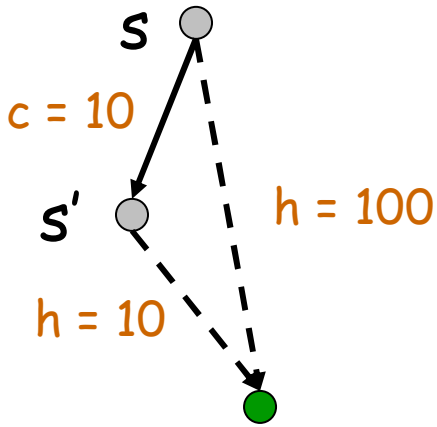


(disuguaglianza triangolare)

- Intuitivamente
  - h(goal) = 0
  - h(s) è una stima che diventa più precisa avvicinandosi al goal
- Fatti
  - Un'euristica h(s) **consistente** è anche **ammissibile**
  - Non vale il viceversa

Funzione euristica consistente (2)

- Euristiche consistenti
  - La tipica violazione si ha per 'eccesso di ottimismo'
  - Nel caso in figura, h(s) è troppo pessimistica (*raro, se h(s) è ammissibile) o h(s') dovrebbe essere ≥ 90

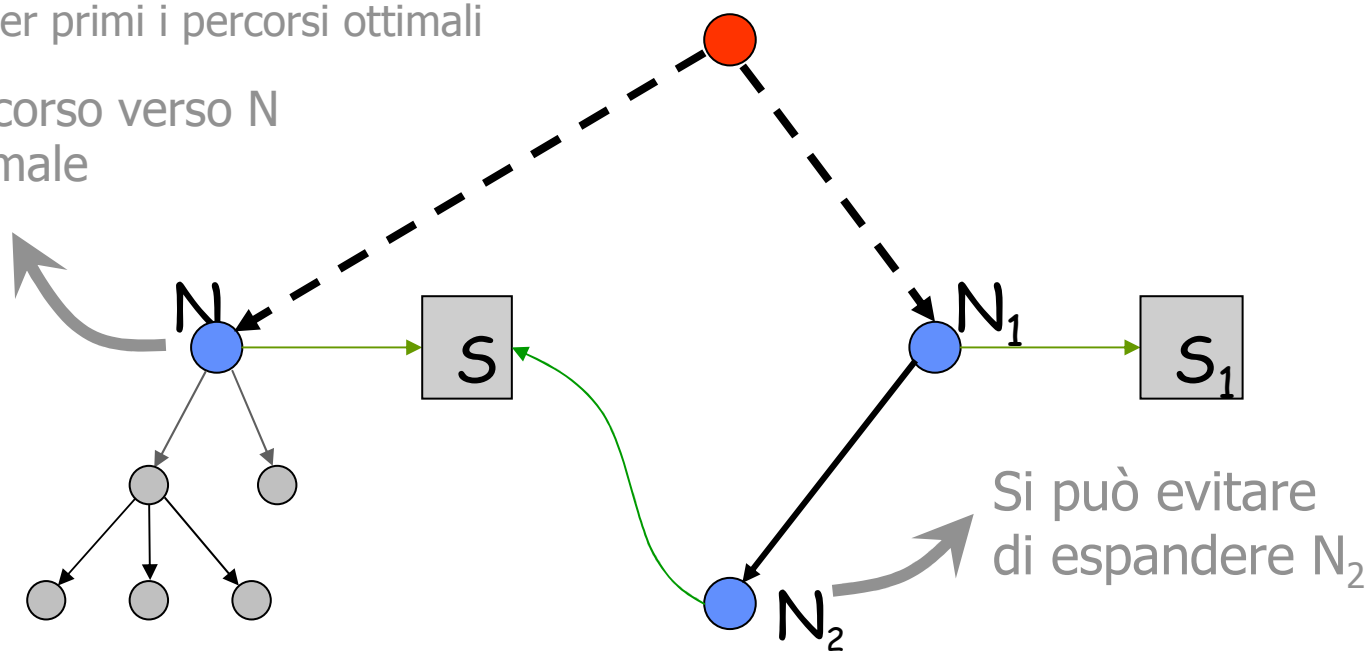


(tipica violazione)

- Esempi:
  - Euristiche consistenti
    - Gioco dell'otto: numero delle parti fuori posto
    - Gioco dell'otto: sommatoria delle distanze (Manhattan) delle parti dalla posizione goal
      - Distanza Manhattan (o a blocchi): ci si muove su una griglia ortogonale senza spostamenti in diagonale
  - Euristiche non consistenti
    - Movimento di un robot su una griglia: distanza Manhattan se il robot può muoversi anche in diagonale

Algoritmo A* con euristica consistente

- Si può dimostrare che
  - L'Algoritmo A* che utilizza un euristica consistente
    - è completo e ottimale
    - rimane tale anche se si evita l'espansione degli stati già visitati
  - In pratica
    - Esplora per primi i percorsi ottimali



Complessità di A*

- Con euristica consistente e spazio degli stati finito
  - Numero di nodi espansi: O(n) dove n è il numero degli stati
  - Complessità di calcolo: $O(n^2)$
  - Complessità di memoria: O(n)

Metodi di ricerca locale (1)

- Definizione (informale)
  - (Si applicano ad euristiche qualsiasi)
  - Si esplora lo spazio degli stati senza costruire un albero
  - Mantenendo solo la rappresentazione di una (singola) posizione corrente
  - In generale:
    - In ogni stato s si considerano Successors(s) e si sceglie in base ad h(s)
    - Sono metodi non completi e non ottimali
      - (salvo in casi particolari, p.es. se lo stato 'riassume' il percorso: qualunque spazio degli stati aciclico può essere espanso)
    - Complessità di memoria ridottissima
    - Somiglianza con i metodi di ottimizzazione nel campo del continuo

Metodi di ricerca locale (2)

- Esempi
  - Discesa ripida (*Steepest descent*)
    - (~ il metodo del gradiente)
    - Ad ogni passo, si sceglie in Successors(s) lo stato con h(s') minore
  - Discesa Monte Carlo
    - Ad ogni passo, si sceglie s' a caso in Successors(s)
    - Se $h(s') \leq h(s)$ si accetta s' e si continua
    - Altrimenti, si accetta s' con probabilità $\sim e^{(-\Delta h / T)}$

      $\Delta h = h(s') - h(s)$

      T è un valore, detto **temperatura**
    - Altrimenti, si sceglie un altro s'
  - *Simulated annealing* (= tempratura simulata)
    - Come la Discesa Monte Carlo
    - La temperatura T si abbassa gradualmente con il progredire dell'algoritmo