

## Ricerca locale

Maria Simi  
2011-2012

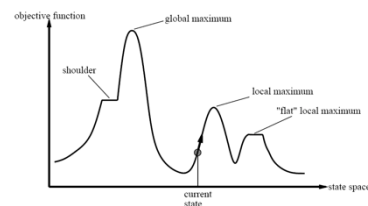
## Assunzioni sui problemi

- Gli algoritmi visti esplorano gli spazi di ricerca alla ricerca di un goal e restituiscono un *cammino soluzione*
- Ma a volte lo stato goal è la soluzione del problema.
- Gli algoritmi di *ricerca locale* sono adatti per problemi in cui:
  - La sequenza di azioni non è importante: quello che conta è lo stato goal
  - Tutti gli elementi della soluzione sono nello stato ma alcuni vincoli sono violati. Es. le regine nella versione a stato completo

## Algoritmi di ricerca locale

- Non sono sistematici
- Tengono traccia solo dello stato corrente e si spostano su stati adiacenti
- Non si tiene traccia dei cammini
- Efficienti in occupazione di memoria
- Utili per risolvere problemi di ottimizzazione
  - lo stato migliore secondo una funzione obiettivo
  - lo stato di costo minore

## Panorama dello spazio degli stati



- Uno stato ha una posizione sulla superficie e una altezza che corrisponde al valore della f. di valutazione
- Un algoritmo provoca movimento sulla superficie
- Trovare l'avvallamento più basso o il picco più alto

## Ricerca in salita (*Hill climbing*)

- Ricerca locale *greedy*
- Vengono generati i successori e valutati; viene scelto un nodo che migliora la valutazione dello stato attuale (non si tiene traccia degli altri):
  - il migliore (salita rapida) → Hill climbing a salita rapida
  - uno a caso → Hill climbing stocastico
  - il primo → Hill climbing con prima scelta
- Se non ci sono stati successori migliori l'algoritmo termina con fallimento

## L'algoritmo Hill climbing

```
function Hill-climbing (problema)
  returns uno stato che è un massimo locale
  nodo-corrente = CreaNodo(problema.Stato-iniziale)
  loop do
    vicino = il successore di nodo-corrente di valore più alto
    if vicino.Valore ≤ nodo-corrente.Valore then
      return nodo-corrente.Stato // interrompe la ricerca
    nodo-corrente = vicino
```

- Nota: si prosegue solo se il vicino è migliore dello stato corrente

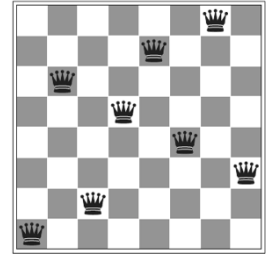
## Il problema delle 8 regine

- $h$ : numero di coppie di regine che si attaccano a vicenda (valore 17)
- I numeri sono i valori dei successori
- Tra i migliori (valore 12) si sceglie a caso

18	12	14	13	13	12	14	14
14	16	13	15	12	14	12	16
14	12	18	13	15	12	14	14
15	14	14	♙	13	16	13	16
♙	14	17	15	♙	14	16	16
17	♙	16	18	15	♙	15	♙
18	14	♙	15	15	14	♙	16
14	14	13	17	12	14	12	18

## Un massimo locale

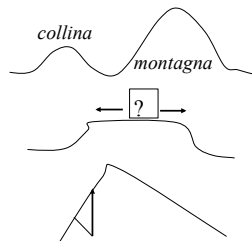
- $h = 1$
- Tutti gli stati successivi peggiorano la situazione
- Per le 8 regine Hill-climbing si blocca l'86% delle volte
- In media 4 passi



## Problemi con Hill-climbing

Se la  $f$ . è da ottimizzare i picchi sono massimi locali o soluzioni ottimali

- Massimi locali
- Pianori o spalle
- Crinali (o creste)



## Miglioramenti

1. Consentire un numero limitato di mosse *lateral*
  - L'algoritmo sulle 8 regine ha successo nel 94%, ma impiega in media 21 passi
2. Hill-climbing stocastico: si sceglie a caso tra le mosse in salita
  - converge più lentamente ma a volte trova soluzioni migliori
3. Hill-climbing con prima scelta
  - può generare le mosse a caso ed essere più efficace quando i successori sono molti

## Miglioramenti (cont.)

4. Hill-Climbing con *riavvio casuale* (*random restart*): ripartire da un punto scelto a caso
  - Se la probabilità di successo è  $p$  saranno necessarie in media  $1/p$  ripartenze per trovare la soluzione (es. 8 regine,  $p=0.14$ , 7 iterazioni)
  - Hill-climbing con random-restart è tendenzialmente completo (basta insistere)
  - Per le regine: 3 milioni in meno di un minuto!
  - Se funziona o no dipende molto dalla forma del panorama degli stati

## Tempra simulata

- L' algoritmo di tempra simulata (Simulated annealing) [Kirkpatrick, Gelatt, Vecchi 1983] combina hill-climbing con una scelta stocastica (ma non del tutto casuale vhe ` poco efficiente...)
- Analogia con il processo di tempra dei metalli in metallurgia

## Tempra simulata

- Ad ogni passo si sceglie un successore a caso:
  - se migliora lo stato corrente viene espanso
  - se no (caso in cui  $\Delta E = f(n') - f(n) < 0$ ) quel nodo viene scelto con probabilità  $p = e^{\Delta E/T}$  [ $0 \leq p \leq 1$ ]

[Si genera un numero casuale tra 0 e 1: se questo è  $< p$  il successore viene scelto, altrimenti no]
- T *descresce* col progredire dell'algoritmo (quindi anche  $p$ ) secondo un piano definito (valore iniziale e decremento sono parametri).

## Tempra simulata: analisi

- La probabilità di una mossa in discesa diminuisce col tempo e l'algoritmo si comporta sempre di più come Hill Climbing. Se T viene decrementato abbastanza lentamente siamo sicuri di raggiungere la soluzione ottimale.
- Analogia col processo di tempra dei metalli
  - T corrisponde alla temperatura
  - $\Delta E$  alla variazione di energia
- Valori per T determinati sperimentalmente: il valore iniziale di T è tale che per valori medi di  $\Delta E$ ,  $p = e^{\Delta E/T}$  sia all'incirca 0.5

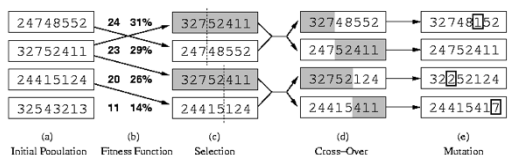
## Ricerca *local beam*

- Si tiene traccia di  $k$  stati anziché uno solo
- Ad ogni passo si generano i successori di tutti i  $k$  stati
  - Se si trova un goal ci si ferma
  - Altrimenti si prosegue con i  $k$  migliori tra questi
- Nella variante *local beam stocastica*, si scelgono  $k$  successori a caso con probabilità maggiore per i migliori (selezione naturale).

## Algoritmi genetici (AG)

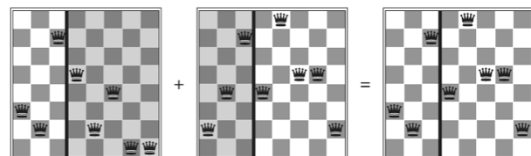
- Popolazione** iniziale:  $k$  stati/individui generati casualmente
- Ogni individuo è rappresentato come una stringa
  - Esempio: 24 bit o "24748552" stato delle 8 regine
- Gli individui sono valutati da una funzione di *fitness*
  - Esempio: n. di coppie di regine che non si attaccano
- Si selezionano gli individui per gli "accoppiamenti" con una probabilità proporzionale alla fitness
- Le coppie danno vita alla generazione successiva, che dovrebbe essere migliore

## Esempio



- Per ogni coppia viene scelto un punto di *cross-over* e vengono generati due figli scambiandosi pezzi
- Viene infine effettuata una *mutazione* casuale che dà luogo alla prossima generazione.

## Nascita di un figlio



- Le parti chiare sono passate al figlio
- Le parti grigie si perdono
- Se i genitori sono molto diversi anche i nuovi stati sono diversi