Risoluzione automatica di problemi

Sistemi distribuiti LS

Prof. Andrea Omicini

A.A. 2003-2004

Problemi del risolvere problemi

- *Goal*
  - *organizzare il comportamento del problema attorno al goal*
  - *rappresentare il goal*
    - *insieme di stati del mondo desiderabili*
- *Problema*
  - *rappresentare il problema*
    - *dato un goal, quali azioni e quali stati rilevanti considerare*
- *Quali azioni compiere?*
  - *in generale, non c'è una singola azione che ci porta al soddisfacimento del goal*
  - *come stabilisco la/le sequenza/e di azioni che mi portano al goal?*

Ricerca

- *Esaminare le diverse sequenze di azioni a disposizione dell'agente*
- *Stabilire quale scegliere*
  - *possibilmente la migliore*
  - *detta la **soluzione***
- *Segue la **esecuzione** della sequenza scelta*

Quale ambiente?

- *Statico*
  - *nel senso che non deve cambiare*
    - *durante la ricerca*
    - *per la parte rilevante ai fini dell'agente*
- *Stato iniziale osservabile o noto*
- *Discreto*
  - *per enumerare le possibili sequenze di stati / azioni*
- *Deterministico*
  - *per conoscere esattamente l'effetto delle azioni*
- *Nota*
  - *per i controllisti, è un sistema in catena aperta*

Definizione del problema

- *Stato iniziale*
  - *modello e percezione*
- *Azioni ammissibili*
  - *funzione successore (o relazione)*
  - *pre- e post- condizioni*
- *Spazio degli stati (raggiungibili)*
  - *dato lo SI e le AA*
  - *grafo*
    - *cammino tra due stati*
- *Test di soddisfacimento del goal*
- *Funzione di misura di costo del cammino*
  - *dal costo di un singolo passo*

Formulazione del problema

- *Scelta della descrizione dello stato*
  - *procedimento di **astrazione***
- *Astrazione anche nella rappresentazione delle azioni*
  - *coerentemente con le scelte sullo stato*
- *Esempio: scelta di una traiettoria di viaggio*

Categorie di esempi e problemi

- *Esempi "toy"*
  - *mondi fittizi, intesi solo a illustrare approcci e soluzioni in maniera semplificata e chiara*
- *Esempi "real world"*
  - *problemi di interesse reale*
- *Un approccio dovrebbe essere*
  - *spiegato con un toy problem*
  - *validato con uno o più real world problem*

Toy problems

- *Mondo dell'aspirapolvere*
  - *stati: 2 piastrelle, pulite o sporche (8 stati)*
  - *iniziale: uno qualunque*
  - *azioni: destra, sinistra, aspira*
- *Il 15 (o 8)*
  - *vari stati finali accettabili*
- *Le 8 regine*
  - *diverse descrizioni stati ammissibili*
  - *100 regine: 10e400 o 10e52 stati*
    - *non si risolvono così…*

Real World Problems

- *Route-finding problems*
- *Touring problems*
  - *TSP*
- *VLSI Layout*
- *Robot Navigation*
- *Protein Design*
- *Internet Searching*

L’albero delle soluzioni

- *Costruzione*
  - *Stato iniziale: radice dell’albero*
  - *Azioni ammissibili: espansione di un nodo*
  - *Ricerca della soluzione: quale foglia espandere?*
    - *Strategia di ricerca*
- *Proprietà*
  - *nodo: stato, genitore, azione, costo, profondità*
  - *distinzione tra nodo e stato*
    - *tra albero di ricerca e spazio degli stati*
  - *insieme delle foglie di un albero (frontiera)*
    - *nodi generati ma non espansi*

Prestazioni

- *4 parametri*
  - *completezza*
  - *ottimalità (della soluzione)*
  - *complessità temporale*
    - *spesso misurata in termini di nodi generati*
  - *complessità spaziale*
    - *misure di complessità: dimensione spazio stati*
      - *oppure b (branching factor), d (profondità shallowest goal), m (massimo cammino nello spazio degli stati)*
- *Costo della ricerca: tempo + spazio*
- *Costo totale: costo ricerca + costo del cammino trovato*

Strategie non informate

- *Indipendenti dal dominio di ricerca*
  - *sanno generare l'albero, e distinguere stati goal da non*
- *Tipi*
  - *ricerca in ampiezza*
    - *ricerca a costo uniforme*
  - *ricerca in profondità*
  - *ricerca in profondità limitata*
  - *ricerca iterativa in profondità*
  - *ricerca bidirezionale*

Ricerca in ampiezza

- *Prima si espande la radice*
  - *poi i successori di primo livello,*
  - *poi quelli di secondo,*
  - *ecc.*
- *La ricerca ha sempre un livello*
  - *coda della frontiera di quel livello*
- *Ricerca completa*
  - *il goal shallowest però non è necessariamente l'ottimale*
  - *Costo: $O(b^{d+1})$ nodi generati, complessità esponenziale*
    - *es: b=10, d=14, 10e15 nodi*
    - *solo le piccole istanze possono essere davvero risolte*

Ricerca a costo uniforme

- *Espande prima il nodo con il cammino meno costoso*
  - *se il costo di ogni passo è 1, uguale alla ricerca in profondità*
  - *se c'è un cammino a costo nullo che riporta allo stesso stato, loop*
  - *se no, è una strategia ottimale*
- *Costo: se C* è il costo della soluzione ottimale*
  - ϵ *costo minimo azione*
  - *complessità caso peggiore* $O(b^{C/\epsilon})$
    - *tende a esplorare spazi ampi con piccoli passi*
    - *se passi costano uguale, sempre* $O(b^{d})$

Ricerca in profondità

- *Espande sempre il nodo più profondo del profilo*
  - *ovviamente non completa*
  - *ma chiede poco spazio (bm+1)*
    - *confronto con breadth-first*
- *Variante: backtracking search*
  - *espande sempre un solo successore per volta*
    - *serve solo O(m) di memoria*
      - *teniamo traccia dei punti di scelta*
  - *molto buono per problemi enormi*
- *Problema: può finire facilmente nel ramo sbagliato…*

Ricerca in profondità limitata

- *Aggiungiamo un limite l alla ricerca in profondità*
  - *se $l < d$, ulteriore fonte di incompletezza*
    - *si può a volte calcolare l dal problema*
  - *comunque, evitiamo i rami infiniti*
- *Complessità temporale $O(b^l)$*
- *Complessita spaziale $O(bl)$*
- *Due risposte*
  - *fallimento (nessuna soluzione)*
  - *cutoff (nessuna soluzione entro l)*

Ricerca iterativa in profondità

- *Misto ampiezza / profondità limitata*
  - *si ripete la ricerca in profondità aumentando l da 0 di 1 per volta*
  - *un sacco di ripetizioni, sembra assurda…*
  - *ma le ripetizioni sono sempre trascurabili o dello stesso ordine del "nuovo"*
- *Complessità spaziale tipo ricerca in profondità*
- *Completezza come ricerca in ampiezza*
  - *e efficienza superiore*
- *La migliore strategia non informata quando lo spazio è grande, e d non è noto*

Ricerca bidirezionale

- *Idea: parto contemporaneamente dallo stato inziale e da quello finale, e cerco di trovare un punto di incontro*
  - *ossia, un'intersezione tra le due frontiere*
  - $O(b^{d/2})$
- *Problema: quando lo stato finale non è uno o descrtitto estensivamente*
  - *es. scacco matto*

Stati ripetuti

- *Gli algoritmi senza storico possono ripetere ricerche già fatte*
- *Gli algoritmi che ricordano, in realtà visitano grafi*
  - *ricerca su grafi: stato già raggiunto, nuovo cammino scartato*
    - *nel caso di strategie ottimali viste, non è un problema*

Ricerca con informazione parziale

- *Cosa succede se l'informazione è incompleta?*
  - *problemi senza sensori: stato iniziale non determinato completamente*
    - *si ragiona su insiemi di stati in cui si è e si può andare*
  - *problemi contingenti: ambiente parzialmente osservabile, o azioni incerte*
    - *controllare dopo ogni azione, e ri-deliberare*
  - *problemi di esplorazione: nulla è noto, esplorare*

Ricerca informata

- *Alcuni problemi sono troppo complessi per essere risolti con strategie generali, in pratica*
- *Spesso, la conoscenza del dominio del problema consente di trovare soluzioni*
  - *magari non ottimali*
  - *ma efficaci / rispondenti ai requisiti*
- *Necessità di strategie che sfruttano la conoscenza sul dominio*
  - *strategie "informate"*

Strategie di ricerca informata

- *Idea: espandere il nodo "migliore" della frontiera*
  - *secondo la nostra conoscenza del dominio*
    - *ovviamente è un "expected best": f(n) funzione di valutazione*
- *Chiave: funzione euristica di valutazione dei nodi*
  - *h(n) è il costo stimato da un nodo n a un goal*
  - *h(goal) = 0*
- *Tipi*
  - *greedy best-first search*
  - *A* search*
- *Il meta-livello*

Greedy best-first search

- $f(n) = h(n)$
  - *esempio di euristica: distanza km tra due città*
- *non è completa, efficienza bassina*
  - *ma una buona euristica può fare miracoli*

A* search

- *Strategia più nota*
- *f(n) = g(n) + h(n)*
  - *g(n) costo per raggiungere il nodo n*
  - *così considero in pratica il costo totale atteso, non solo quello mancante…*
  - *se h è una euristica ammissibile*
    - *cioè non sovrastima mai il costo, è ottimistica*
  - *allora la strategia è ottimale*
    - *f non è mai una sovrastima del costo effettivo*
  - *e può essere anche completa*
  - *… ma non andiamo oltre*

Meta-livello

- *Si può fare meglio?*
  - *può l'agente imparare strategie nuove?*
  - *sì, se si mette a ragionare al meta-livello*
    - *suk meta-spazio delle soluzioni, e non sul livello oggetto*
- *Ovviamente sono necessari algoritmi di apprendimento di metalivello*

Funzioni euristiche

- *Produrre euristiche*
  - *per molti problemi pratici, il problema primario è produrre una euristica sufficientemente efficace*
- *Non approfondiamo*
  - *è una cosa "artistica", spesso*
  - *ci sono criteri generali*
    - *effective branching factor*
    - *dominanza relativa tra euristiche*
  - *ma c'è anche la fattorizzazione della buona vecchia esperienza*
  - *magari seguita da una prova teorica di dominanza…*

Algoritmi di ricerca locale e problemi di ottimizzazione

- *Molti problemi non richiedono il cammino verso la soluzione*
  - *es. 8 regine: basta la soluzione*
- *Quindi si generano solo gli stati successivi, senza tenere memoria*
  - *algoritmi di ricerca locale*

Esempi di algoritmi

- *hill-climbing search*
- *simulated annealing search*
- *local beam search*
- *algoritmi genetici*

Ricerca online e esplorazione

- *Approcci offline vs. online*
  - *online: calcolo e azione sono interleaved*
  - *tipicamente: unico modo di fare andare i robot*
- *Online: necessario per l'esplorazione*
  - *ambiente sconosciuto*
- *Idea: può essere risolto solo da un agente che esegue azioni*
  - *es.: esploratore*
    - *non posso sapere il successore di uno stato finché non lo provo*
- *Vari obbiettivi possibili*
  - *minizzare il costo, sapere tutto, non morire, …*

Alcuni criteri

- *Competitive ratio*
  - *costo del cammino effettivo rispetto a quello che l'agente avrebbe compiuto se avesse saputo prima*
    - *a volte è infinito, come in caso di azioni irreversibili che portano in stati sbagliati*
- *Mappa*
  - *l'esplorazione può essere condotta con lo scopo di produrre "mappe" dello spazio*
    - *supponendo safe e stabile l'environment*
    - *un approccio casuale non è detto sia sbagliato…*
  - *la mappa può essere usata al volo o dopo*
- *Apprendimento*
  - *utile dare all'agente la capacità di imparare*