Algoritmi genetici applicati ai problemi di partizionamento su alberi

Luca Rapposelli

26 ottobre 2003

Università degli Studi Roma Tre – Facoltà di Scienze M.F.N. – Corso di Laurea in Matematica

Descrizione del problema

- Abbiamo studiato il **problema di ottimizzazione discreta** relativo al **partizionamento di grafi**
- In particolare: **equipartizione in p componenti connesse di alberi** sulla base di diverse funzioni obiettivo
- Il problema è **NP-completo** su grafi generici; in alcuni casi è NP-completo anche su alberi
- Abbiamo utilizzato la tecnica degli **algoritmi genetici** per produrre una soluzione "buona" in tempi estremamente brevi

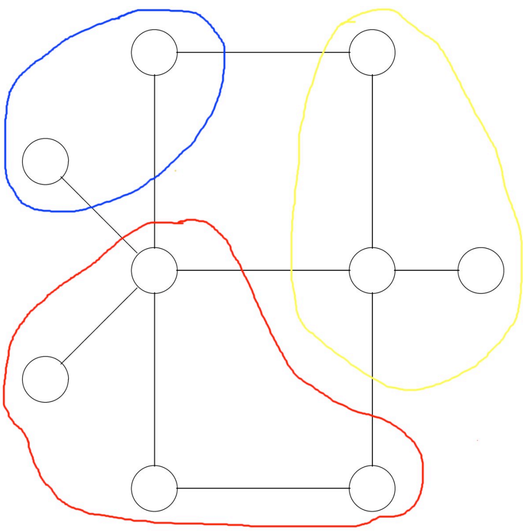
Partizionamento di grafi

- Dato un grafo $G = (V, E)$ una **partizione in** p **componenti** di G è una famiglia $\pi = \{C_1, \ldots, C_p\}$ tale che $C_i \subseteq V$, $C_i \cap C_j = \emptyset$ per $i \neq j$, $\cup_{i=1}^{p} C_i = V$

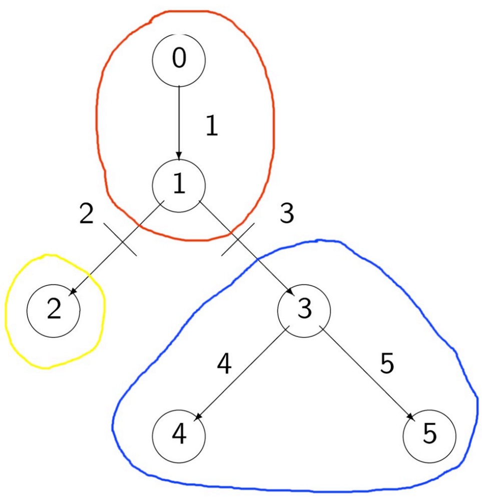
- La partizione è in **componenti connesse** se il sottografo indotto da ogni componente C_i è connesso

- $\Pi(G)$: insieme di tutte le partizioni di G; $\Pi_p(G) \subset \Pi(G)$ insieme di tutte le p-partizioni di G

- Problema di ottimizzazione: trovare una partizione $\pi^* \in \Pi(G)$ tale che il valore di una **funzione obiettivo** calcolata su π^* sia massimo (o minimo)



Partizione di un grafo



Partizione di un albero

Equipartizione di grafi

- Partizionamento di **grafi pesati**, con pesi assegnati ai vertici:
  - Il peso del vertice v_i è $W(v_i) = w_i$ con $w_i \in \mathbb{Z}^+$
  - Il peso della componente C_j è $W(C_k) = \sum_{v_i \in C_k} w_i$
  - Il peso medio di una componente della partizione è $\mu = \frac{1}{p} \sum_{k=1}^{p} W(C_k)$
- **Equipartizione**: partizione in p componenti che aggregano vertici con pesi simili; ovvero partizione in p componenti di peso equivalente
- L'equipartizione di un grafo pesato è un **problema di ottimizzazione**:
  - Minimizzare la funz. obiettivo Norma L_1: $f(\pi) = \sum_{k=1}^{p} |W(C_k) - \mu|$
  - Minimizzare la funz. obiettivo Norma L_∞: $f(\pi) = \max_k |W(C_k) - \mu|$

Evoluzione biologica

- In natura ogni essere vivente possiede un patrimonio genetico rappresentato da **cromosomi**, formati da una sequenza di **geni**
- Se un organismo è **idoneo** a sopravvivere avrà una maggiore aspettativa di vita e dunque avrà una maggiore probabilità di essere **selezionato** per riprodursi e tramandare i propri geni
- La **riproduzione** è la ricombinazione (incrocio) del patrimonio genetico tra una coppia di genitori per la creazione di nuovi discendenti
- La **mutazione** genetica è il cambiamento di uno o più geni in un cromosoma per adattarsi all'ambiente circostante

Algoritmi genetici

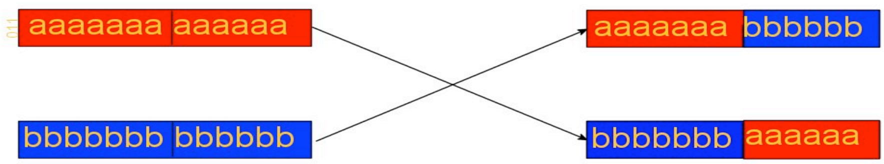
- Si adotta una **codifica** per rappresentare ogni elemento π_i dello spazio delle soluzioni con un **cromosoma** c_i
- La **funzione idoneità** $f(c_i)$ misura la qualità della soluzione rappresentata da un cromosoma c_i (è analoga alla funzione obiettivo)
- Si applicano ripetutamente le operazioni di **selezione**, **incrocio** e **mutazione** per far evolvere una popolazione iniziale arbitraria di cromosomi, fino a raggiungere la soluzione ottima del problema

Operazioni genetiche

- **Selezione**: il cromosoma c_i viene **selezionato** per riprodursi in base alla probabilità $p_s = \frac{f(c_i)}{\sum_{k=1}^{m} f(c_k)}$: sono favoriti i cromosomi c_i con idoneità $f(c_i)$ più elevata
- **Incrocio**: si ricombinano i geni dei cromosomi selezionati per formare due nuove stringhe
- **Mutazione**: è il cambiamento del valore di uno o più geni di un cromosoma; permette di sfuggire da massimi o minimi locali

Incrocio

Cromosomi genitori Cromosomi figli

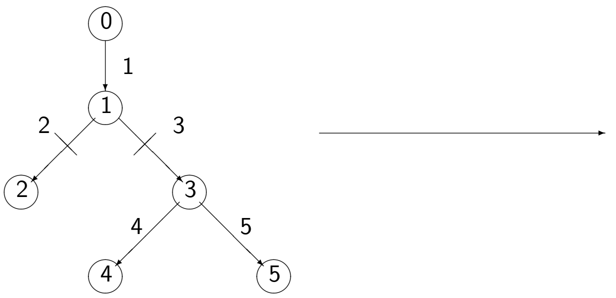


Passi di un algoritmo genetico

1. Genera casualmente una popolazione iniziale di m cromosomi di lunghezza l
2. Calcola la **funzione idoneità** per ogni cromosoma
3. Ripeti, finché non sono stati creati m discendenti, le seguenti operazioni:
   (a) Con probabilità p_s **seleziona due cromosomi** in modo da privilegiare i cromosomi con un alto valore della funzione idoneità
   (b) Con probabilità p_c **incrocia i due cromosomi**; se questo incrocio non avviene i due cromosomi rimangono invariati
   (c) Con probabilità p_m **muta i due cromosomi**; se non succede lascia i cromosomi invariati
4. **Sostituisci** la vecchia popolazione di cromosomi con quella appena generata.
5. Vai al passo 2 se la condizione di stop non è verificata.
6. Stampa la **migliore soluzione** individuata durante l'esecuzione dell'algoritmo

Algoritmi genetici e partizionamento di alberi (1/2)

Codifichiamo una partizione in p componenti di un albero $\pi \in \Pi_p(G)$ con una stringa di $\{0,1\}$, dove ogni gene rappresenta un arco dell'albero, la totalità degli archi forma il cromosoma, il **valore** 0 rappresenta un **arco senza taglio** e il **valore** 1 un **arco con il taglio**



cromosoma (stringa):

$a = 01100$

$a_1 = 0,\ a_2 = 1,\ a_3 = 1,$
$a_4 = 0,\ a_5 = 0$

Algoritmi genetici e partizionamento di alberi (2/2)

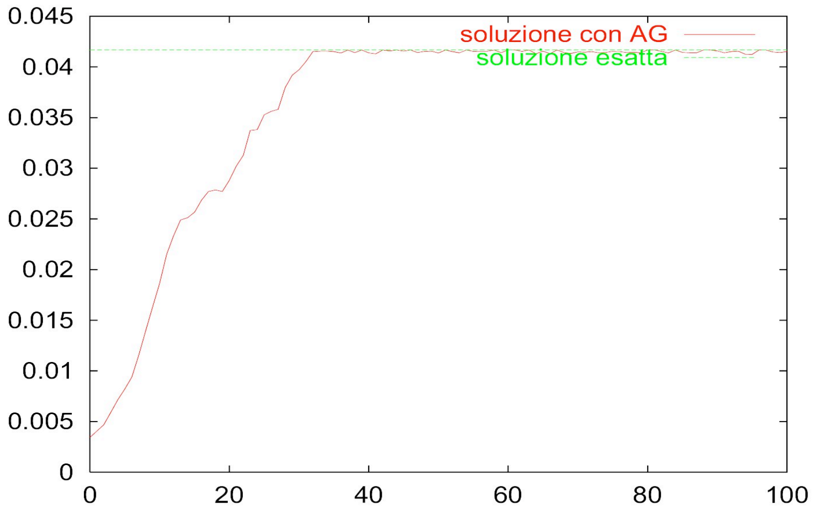
- Per calcolare la **funzione idoneità** si effettua una visita dell'albero calcolando i pesi delle partizioni, per ottenere la norma L_1 o la norma L_∞
- Si **incrociano** due stringhe per avere due nuove partizioni in p componenti connesse
- La **mutazione** equivale allo spostamento di uno o più tagli nell'albero

Risultati sperimentali – Esempio n.1

Dati del problema di ottimizzazione con funzione L_∞:

- Albero con $n = 100$ vertici, partizionato in $p = 10$ componenti, peso dei vertici $w_i \in [1, 10]$
- Popolazione di 200 cromosomi, probabilità di mutazione $p_m = 0.01$ con lo spostamento di massimo tre tagli, incrocio uniforme con $p_c = 1$.
- Risultato ottimo 24 trovato alla generazione 14.

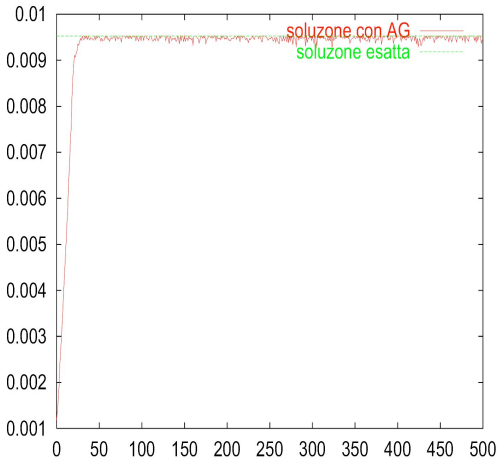
Risultati sperimentali – Esempio n.1



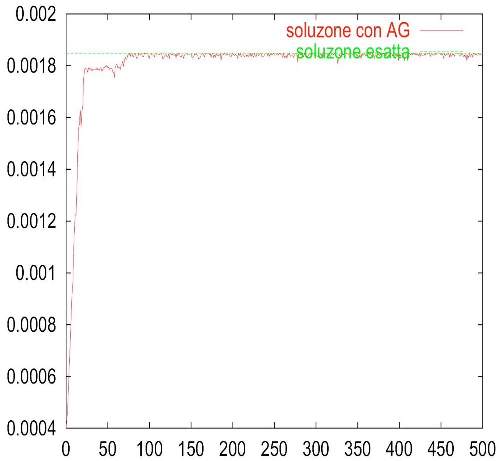
Risultati sperimentali – Esempio n.2

- Albero con $n = 50$ vertici, partizionato in $p = 10$ componenti, peso dei vertici $w_i \in [1, 100]$; uso come funzioni obiettivo la **norma** L_∞ e la **norma** L_1
- Popolazione di 100 cromosomi, probabilità di mutazione $p_m = 0.01$, incrocio uniforme con $p_c = 1$.
- Il risultato ottimo (105) trovato alla generazione 14 per la norma L_∞, e 541 alla generazione 44 per la norma L_1.

Risultati sperimentali – Esempio n.2



norma L_∞



norma L_1