Algoritmi Genetici

con applicazioni ai giochi ed ai problemi di ottimizzazione

Candidato: Daniele Rampoldi
Relatore: Prof. Stuart Coles

Università degli studi di Padova, Facoltà di Scienze Statistiche
Corso di laurea specialistica in Statistica e Informatica
A.A.2005/2006

23 Marzo 2006

Principi di base - L'evoluzione biologica

Gli Algoritmi Genetici sono una famiglia di **tecniche di ottimizzazione** che si ispirano alla teoria della **selezione naturale** di *Charles Darwin* che regola l'evoluzione biologica.

I principi di base che contraddistinguono lo sviluppo filogenetico degli organismi viventi sono:

- l'evoluzione naturale opera sui **cromosomi** degli individui invece che sugli individui stessi, ovvero su di una codifica genetica (il 'genotipo') delle caratteristiche fisiche dell'organismo (il 'fenotipo');
- il processo di selezione naturale favorisce la riproduzione dei cromosomi che hanno dato luogo agli **organismi più efficienti** dal punto di vista adattativo;
- la combinazione dei codici genetici dei due genitori e l'introduzione di piccole mutazioni casuali dà luogo alla **creazione di nuove strutture genetiche**;
- l'evoluzione naturale opera su popolazioni di individui attraverso un **processo ciclico e generazionale**.

Principi di base - Gli algoritmi genetici

Gli **Algoritmi Genetici** operano su di una **popolazione di cromosomi artificiali**.

Il processo riproduttivo consiste nei seguenti passi:

1. creazione casuale della prima generazione;
2. valutazione dei cromosomi artificiali;
3. riproduzione selettiva degli individui migliori;
4. accopiamento casuale dei cromosomi (crossover);
5. mutazione casuale.

Le nuove strutture genetiche vanno quindi a rimpiazzare quelle dei genitori dando luogo ad una nuova generazione di individui.
Il processo riparte dal passo (2) e continua fino a quando nasce un individuo che rappresenta una soluzione accettabile per il problema in esame.

Codifica

Per **codifica genetica** ci si riferisce al tipo di rappresentazione che viene utilizzata per identificare le soluzioni del problema nei cromosomi artificiali.

Il funzionamento dell'algoritmo genetico non è compromesso dal tipo di rappresentazione; essa però dovrebbe basarsi su di un alfabeto ristretto che permetta un'espressione naturale del problema.

Funzione di valutazione

La **funzione di valutazione** serve per giudicare le prestazioni di ciascun cromosoma rispetto al problema che vogliamo risolvere: essa deve fornire un valore numerico per ciascun individuo proporzionale alla bontà della soluzione offerta.

La funzione di fitness può assumere vari gradi di complessità a seconda delle conoscenze disponibili: l'unico requisito è che essa permetta un ordinamento delle stringhe genetiche in base alle loro prestazioni sul problema da risolvere.

Riproduzione selettiva

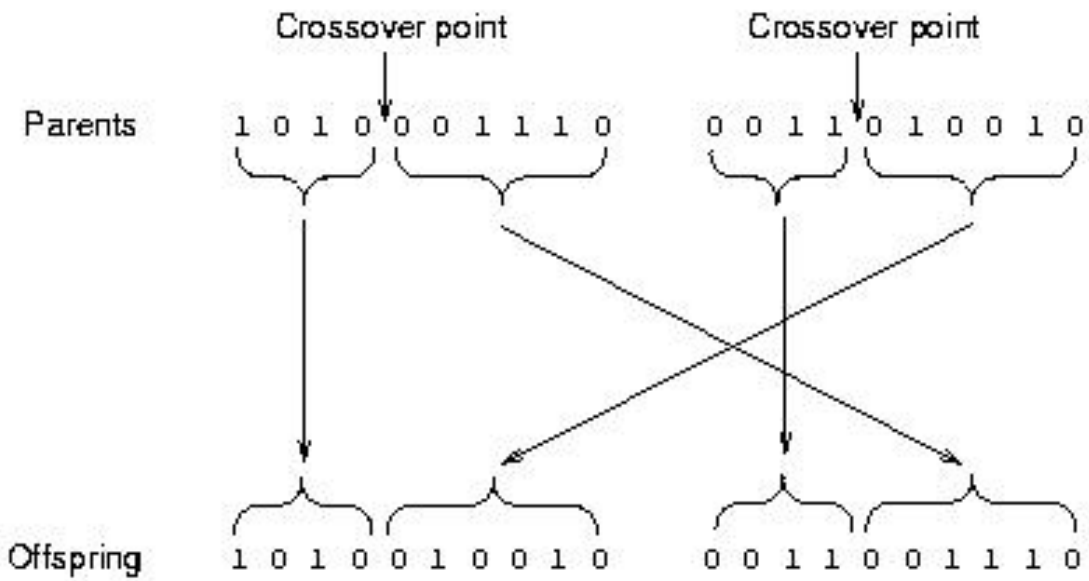
Il processo di **riproduzione selettiva** consiste nella creazione probabilistica di un numero di copie di ciascuna stringa proporzionale al valore di fitness ottenuto dal cromosoma corrispondente.

Ricopiare ciascuna stringa in proporzione al proprio valore di fitness significa che le stringhe che hanno riportato un valore di fitness maggiore avranno una probabilità maggiore di produrre uno o più figli.

Crossover

Le nuove stringhe così create vengono disposte a coppie in modo casuale e sottoposte all'azione dell'operatore di **ricombinazione genetica** (**crossover**).

Per ciascuna delle coppie selezionate viene scelto un punto d'incrocio casuale attorno al quale avviene uno scambio reciproco di materiale genetico.



Mutazione

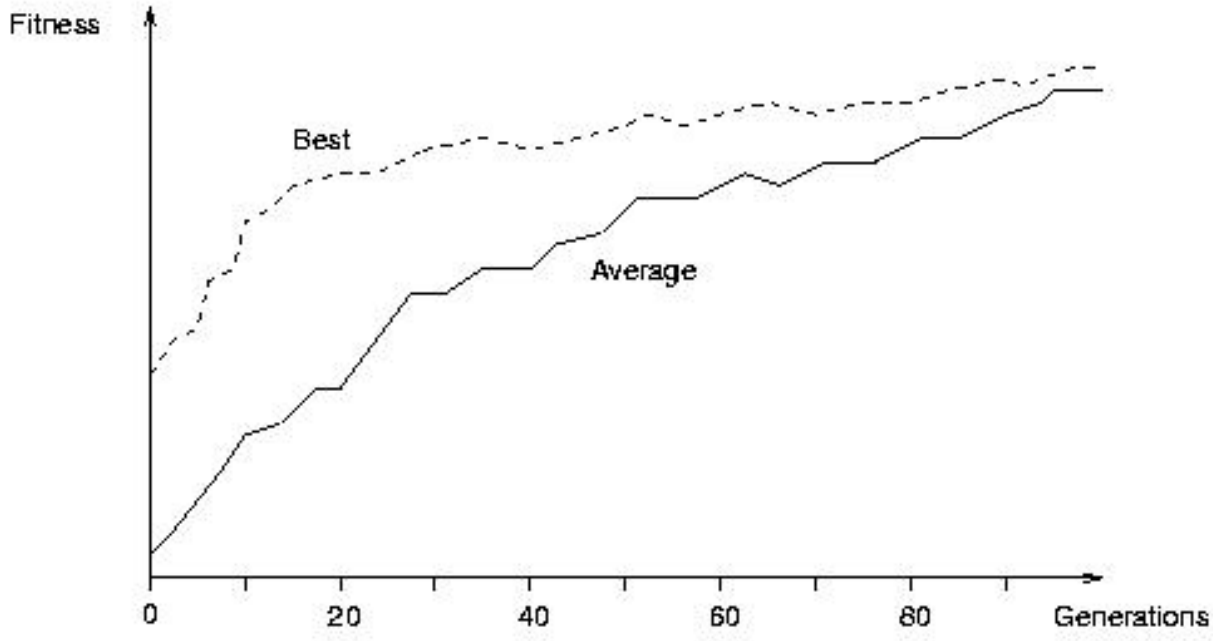
Infine tutte le stringhe della popolazione subiscono un processo di **mutazione**: ciascun elemento del cromosoma cambia il proprio valore in base ad una probabilità p_m.

La teoria tradizionale ritiene che il crossover sia più importante della mutazione per quanto riguarda la rapidità nell'esplorare lo spazio di ricerca; la mutazione porta un po' di 'casualità' in essa in modo che nessun punto nello spazio abbia probabilità nulla di essere esaminato.

Convergenza

La nuova popolazione di stringhe genetiche rimpiazza parzialmente o completamente le vecchie stringhe. Il processo di decodifica, valutazione, riproduzione selettiva, incrocio e mutazione riprende ciclicamente per parecchie generazioni fino a quando viene ottenuta una stringa che codifica una soluzione soddisfacente.

Se l'algoritmo genetico è correttamente implementato, la popolazione evolverà nel susseguirsi delle generazioni in modo che il fitness del miglior individuo e la media in ogni generazione cresca verso l'ottimo globale.



Un primo esempio

Il problema consiste nel trovare i valori della variabile x che massimizzano la funzione potenza $\Phi(x) = x^2$, dove x può variare tra 0 e 31.

Il primo passo consiste nella scelta della rappresentazione genetica: utilizziamo un codice binario e stringhe di lunghezza 5 che ci permettono di codificare numeri (interi) da 0 ('00000') a 31 ('11111').

Num. Stringa	**Popolaz.**	**Decod. x**	**Valutaz.** $F(x)=x^2$	**Prob. di riprod.** $P_i=\frac{f_i}{\sum_i^N f_i}$	**Num. figli**	**Num. estraz.**
1	01101	13	169	0,14	0,58	1
2	11000	24	576	0,49	1,97	2
3	01000	8	64	0,06	0,22	0
4	10011	19	361	0,31	1,23	1
somma			1170	1,00	4,00	4
media			293	0,25	1,00	1
massimo			576	0,49	1,97	2

Un primo esempio - 2

Accoppiam. con crossover (!)	**Incrocio** p_c=*1,0* (tutti)	**Mutazione** p_m=*0,001* (nessuna)	**Nuova popolaz.**	**Decod. x**	**Valutaz.** $\Phi(x) = x^2$
0110!1	01100	01100	01100	12	144
1100!0	11001	11001	11001	25	625
11!000	11011	11011	11011	27	729
10!011	10000	10000	10000	16	256
somma					1754
media					439
massimo					729

Sia il valore medio che il valore massimo di fitness sono superiori al valore della generazione iniziale.

Gli schemi

La teoria degli Algoritmi Genetici ipotizza che essi funzionino scoprendo, favorendo e ricombinando buoni **blocchi costitutivi** di soluzioni.

Uno **schema** è una stringa i cui possibili elementi sono quelli dell'alfabeto genetico utilizzato per la codifica, con l'aggiunta del simbolo '*', che ha il ruolo di carattere jolly.

Uno schema rappresenta quindi un insieme di stringhe genetiche. Ad esempio, se si utilizza il codice binario come alfabeto genetico, lo schema '1111*' rappresenta la stringa '11111' e '11110'.

Mentre l'algoritmo genetico valuta esplicitamente l'idoneità delle stringhe della popolazione attuale, in realtà sta stimando implicitamente l'idoneità media di un numero assai maggiore di schemi, dove l'**idoneità media di uno schema** è definita come la media delle idoneità di tutte le possibili istanze di quello schema.

Per esempio, in una popolazione di n stringhe generate casualmente, mediamente la metà delle stringhe sarà istanza di (1,*,*, ..., *,*) e metà sarà istanza di (0,*,*, ..., *,*). La valutazione delle circa $n/2$ stringhe che sono istanze di (1,*,*, ..., *,*) fornisce una stima dell'idoneità media di quello schema.

Ordine e lunghezza di definizione

L'**ordine** di uno schema S, denotato con $o(S)$, è il numero di posizioni definite, ovvero (nel caso del codice binario) il numero di 0 e di 1 presenti.

$o(S)$ = (lunghezza totale dello schema) - (numero di caratteri jolly)

L'ordine definisce la specificità di uno schema ed è utile per calcolare la probabilità di sopravvivenza di uno schema alla mutazione.

La **lunghezza di definizione** dello schema S, denotata con $\delta(S)$, è la distanza tra la prima e l'ultima posizione fissata nella stringa. Essa definisce la compattezza delle informazioni contenute in uno schema. Si osservi che uno schema con una sola posizione fissata ha lunghezza di definizione uguale a zero.

A titolo di esempio si consideri la lunghezza di definizione e l'ordine dei seguenti schemi di dieci bits:

i	S_i	$o(S_i)$	$\delta(S_i)$
1	***001*110	6	6
2	****00**0*	3	4
3	11101**001	8	9

Teorema degli schemi

Denotiamo con $\xi(S, t)$ il **numero di stringhe** in una popolazione al tempo t che siano rappresentate dallo schema S.

Inoltre possiamo definire *eval(S,t)* come la **fitness media** di tutte le stringhe della popolazione che sono istanza dello schema S al tempo t.

Notiamo quindi che per ogni stringa istanza dello schema S la probabilità della sua selezione è (in media) $\frac{eval(S,t)}{F(t)}$, con $F(t)$ fitness totale dell'intera popolazione al tempo t.

Teorema degli schemi - Equazione di crescita

Dopo la selezione ci aspettiamo di avere $\xi(S, t+1)$ stringhe istanza dello schema S. Il numero atteso di stringhe istanza di S al tempo $t+1$ è pari a:

$$\xi(S, t+1) = \xi(S, t) \cdot n \cdot \frac{eval(S, t)}{F(t)}$$

Ma l'idoneità media della popolazione è $\overline{F(t)} = \frac{F(t)}{n}$. Quindi

$$\xi(S, t+1) = \xi(S, t) \cdot \frac{eval(S, t)}{\overline{F(t)}}$$

Questa formula rappresenta l'equazione della crescita riproduttiva dello schema.

In pratica, se pensiamo che lo schema S sia superiore alla media, in percentuale, di ϵ, cioè:

$$eval(S, t) = \overline{F(t)} + \epsilon \cdot \overline{F(t)}$$

allora otteniamo:

$$\xi(S, t) = \xi(S, 0) \cdot (1+\epsilon)^t$$

Teorema degli schemi - Crossover

In una stringa di lunghezza l, la posizione di incrocio (crossover) è selezionata uniformemente tra le $l-1$ possibili posizioni. Questo implica che la probabilità di distruzione di uno schema è:

$$p_d(S) = \frac{\delta(S)}{l-1}$$

Solo alcuni cromosomi subiscono il processo di incrocio, con probabilità selettiva di incrocio p_c. Inoltre, anche se la posizione di incrocio è fissata in modo tale da rompere uno schema, c'è ancora una possibilità di sopravvivenza data dalla presenza di sotto-schemi uguali in due cromosomi differenti:

$$p_s(S) \geq 1 - p_c \cdot \frac{\delta(S)}{l-1}$$

Gli effetti combinati della selezione e dell'incrocio ci danno una nuova forma dell'equazione di crescita dello schema riproduttivo.

$$\xi(S, t+1) \geq \xi(S, t) \cdot \frac{eval(S, t)}{\overline{F(t)}} \cdot \left[1 - p_c \cdot \frac{\delta(S)}{l-1}\right]$$

Teorema degli schemi - Mutazione

La probabilità di mutazione di un singolo bit è p_m; quindi la probabilità di sopravvivenza di ogni bit è $(1 - p_m)$. Inoltre ogni singola mutazione è indipendente dalle altre mutazioni. Quindi la probabilità di uno schema di sopravvivere alla mutazione è:

$$p_s(S) = (1-p_m)^{o(S)}$$

Gli effetti combinati di selezione, incrocio e mutazione ci danno una nuova forma dell'equazione di crescita dello schema riproduttivo:

$$\xi(S, t+1) \geq \xi(S, t) \cdot \frac{eval(S, t)}{\overline{F(t)}} \cdot \left[1 - p_c \cdot \frac{\delta(S)}{l-1}\right] \cdot (1-p_m)^{o(S)}$$

Il risultato finale dell'equazione di crescita può essere enunciato come Teorema degli Schemi.

Teorema degli schemi

'Schemi con lunghezza di definizione breve, di basso ordine e con fitness superiore alla media subiscono una crescita esponenziale nelle generazioni successive di un algoritmo genetico.'

Affinché la procedura di isolamento e combinazione degli schemi risulti in un graduale miglioramento delle soluzioni offerte è importante che la rappresentazione genetica sia appropriata al problema da risolvere.

Vi sono due regole di base per scegliere una rappresentazione adeguata: la rappresentazione dovrebbe permettere la presenza di schemi rilevanti corti e dovrebbe basarsi su di un alfabeto ristretto che permetta un'espressione naturale del problema.

Aspetti pratici

Vi sono numerosi aspetti pratici o specifici di problemi particolari.

I più importanti sono:

- Codifiche ad hoc
- Mapping dei valori ridondanti
- Elitismo e Steady-State Reproduction
- 2-Point Crossover
- Crossover Uniforme
- Operatori dinamici

Il Master Mind

Il Master Mind è un gioco di 'code-breaking' (decifrazione di codice).

I giocatori sono: un 'Code Maker' (o 'Mastermind') che definisce una sequenza di colori e la tiene nascosta ed un 'Code Breaker' (o 'Seeker') che deve indovinare la giusta combinazione.

Ogni tentativo del 'Cercatore' è valutato dal 'Mastermind' per mezzo di picchetti neri (ognuno dei quali corrisponde ad un colore indovinato nella giusta posizione) e bianchi (per i colori indovinati, ma mal posizionati).

Strategia

Codifica: consideriamo i colori come i possibili geni e le sequenze di colori come cromosomi.

Funzione di valutazione: utilizziamo la valutazione che il *Master Mind* fa di una certa sequenza.

Riproduzione selettiva: la strategia di ricerca dell'ottimo (cioè della soluzione segreta) avrà come iter la sottomissione al Code Maker di una singola prova alla volta. Di conseguenza, ad ogni passo, l'operatore di selezione dovrà avere come punto di partenza un unico cromosoma, dal quale deriverà la prova successiva.

Crossover: il numero di geni trasmessi al discendente è determinato dalla funzione di valutazione: se nella valutazione della miglior prova abbiamo n picchetti neri e b bianchi, conserveremo n colori nella stessa posizione e b in posizione differente.

Mutazione: il numero di geni mutati è determinato dalla funzione di valutazione: rispetto alla miglior prova saranno modificati $p - n - b$ geni, con p numero di geni in un cromosoma.

Consistenza: scarteremo la prova se risulterà essere non consistente (ovvero se sappiamo con certezza che non porterà alla soluzione).

Un esempio pratico

Supponiamo che il Master Mind scelga come sequenza segreta $(1, 5, 1, 3)$, scegliendo tra sei possibili 'colori'.

Secret code: (1,5,1,3)

\# 1: 3 6 5 3 [1,1]

Estraiamo dalla sequenza tanti colori quanti sono i picchetti neri (uno in questo caso) e li inseriamo in una nuova sequenza nella stessa posizione; estraiamo poi dai rimanenti colori della sequenza tanti colori quanti sono i picchetti bianchi (ancora uno) e li inseriamo in un posizione differente.

Supponiamo per esempio di estrarre $(5, *, *, 3)$, dove indichiamo con il simbolo '*' un gene mancante.

Per completare la sequenza utilizziamo l'operatore di mutazione.

Secret code: (1,5,1,3)

\# 1: 3 6 5 3 [1,1]
\# 2: 5 2 4 3 [1,1]

Un esempio pratico - 2

In maniera analoga otterremo la terza prova:

Secret code: (1,5,1,3)

\# 1: 3 6 5 3 [1,1]
\# 2: 5 2 4 3 [1,1]
\# 3: 5 3 5 1 [0,3]

La terza sequenza risulta essere migliore delle precedenti: essa sarà quindi il nostro nuovo punto di partenza.

Il primo passo è estrarre tre geni e cambiarne la posizione per mezzo dell'operatore di crossover.

Supponiamo di trovare $(1, *, 3, 5)$. Questo sarebbe consistente con la terza prova, ma non con le prime due.

Analogamente la sequenza $(3, *, 1, 5)$ sarà consistente con la prima e la terza prova, ma otterrà un risultato di [0,2] se confrontato con la seconda.

Un esempio pratico - 3

Supponiamo di trovare $(*, 5, 1, 3)$ (consistente con le tre prove); prima di procedere all'estrazione del gene mancante facciamo una 'scrematura' dell'insieme dei possibili colori.

Ci rimarrà solamente un colore, che completerà la sequenza a $(1, 5, 1, 3)$.

Secret code: (1,5,1,3)

\# 1: 3 6 5 3 [1,1]
\# 2: 5 2 4 3 [1,1]
\# 3: 5 3 5 1 [0,3]
\# 4: 1 5 1 3 [4,0]

La nostra ricerca è dunque terminata con la scoperta del codice segreto al quarto tentativo.

Bontà dell'algoritmo

	Numero di prove	
	4x6	**5x8**
Rosu	4.66	5.88
Bento	-	6.86
Merelo	4.132 (sd = 1.00)	5.904 (sd = 0.975)
Temporel - Kovacs	4.64 (sd = 0.857)	5.834 (sd = 0.934)
Codice Alternativo	4.335 (sd = 0.6983)	-
MM Genetico	4.646 (sd = 0.88072)	5.897 (sd = 0.95955)

Dati ottenuti simulando la distribuzione delle possibili sequenze, utilizzando come sequenza segreta tutte le possibili sequenze del tipo $(1, *, *, *)$, o $(1, *, *, *, *)$, con cento possibili semi differenti di generazione casuale.

Il codice sviluppato dovrà valutare mediamente meno sequenze.

Bontà dell'algoritmo - 2

	Numero di valutazioni	
	4x6	**5x8**
Rosu	1295	32515
Bento	-	1029.9
Merelo	279 (sd = 188)	2171 (sd = 1268)
Temporel - Kovacs	41.2 (sd = 49.0)	480.1 (sd = 826.6)
MM Genetico	30.88 (sd = 106.88)	155 (sd = 999.49)