Algoritmi Genetici

Dr. Donato D'Ambrosio
Dr. William Spataro
Prof. Salvatore Di Gregorio

Dipartimento di Matematica & Centro per il Calcolo ad Alte Prestazioni
Università della Calabria, Arcavacata, I-87036 Rende (CS)

EMPEDOCLES RESEARCH GROUP

Cosa sono e come operano gli Algoritmi Genetici

- Gli Algoritmi Genetici (AG) furono proposti da John Holland (Università del Michigan) tra la fine degli anni 60 e l'inizio degli anni 70

- Gli AG (Holland, 1975, Goldberg, 1989) sono algoritmi di ricerca che si ispirano ai meccanismi della selezione naturale e della riproduzione sessuale

- Gli AG simulano l'evoluzione di una popolazione di individui, che rappresentano soluzioni candidate di uno specifico problema, favorendo la sopravvivenza e la riproduzione dei migliori

Codifica

- Si assume che una possibile soluzione per un problema possa essere rappresentata come un set di parametri (detti ***geni***) i quali sono uniti insieme per formare una stringa di valori (spesso chiamata ***cromosoma***).

- Holland per primo ha mostrato, ed è ancora accettato da molti, che l'ideale è usare un alfabeto binario per la stringa. Per esempio, se vogliamo rappresentare una funzione di due variabili reali, ***f(x,y)***, possiamo rappresentare la prima variabile con un numero binario di 10 bit, la seconda variabile con un numero binario di 15 bit . Il nostro cromosoma conterrà allora due geni, e consisterà di 25 cifre digitali.

- In termini genetici, l' insieme dei parametri rappresentati da un particolare cromosoma è chiamato ***genotipo*** (sequenza di bit). Il genotipo contiene le informazioni richieste per costruire un organismo (soluzione candidata) che è chiamato ***fenotipo***.

Codifica binaria

- La codifica binaria e, probabilmente, la piu utilizzata nelle applicazioni pratiche, sia per motivi storici, sia perché su essa sono stati derivati i risultati teorici più importanti.
- La struttura dati utilizzata è un vettore di bit di lunghezza ***l***, cui corrisponde uno spazio di ricerca di 2^l possibili soluzioni.
- L'uso della codifica binaria richiede la specificazione di una funzione che decodifichi il genotipo, o parti di esso, nel corrispondente fenotipo.
- Per esempio, la seguente equazione decodifica un genotipo binario ***g*** di lunghezza ***l*** nel corrispondente valore ***x*** .

$$x = x_{min} + \frac{x_{max} - x_{min}}{2^l - 1}\left(\sum_{i=1}^{l} g[i]^{l-i}\right)$$

- dove x_{min} e x_{max} sono rispettivamente il minimo e massimo valore che puo assumere ***x*** .

Codifica grigia

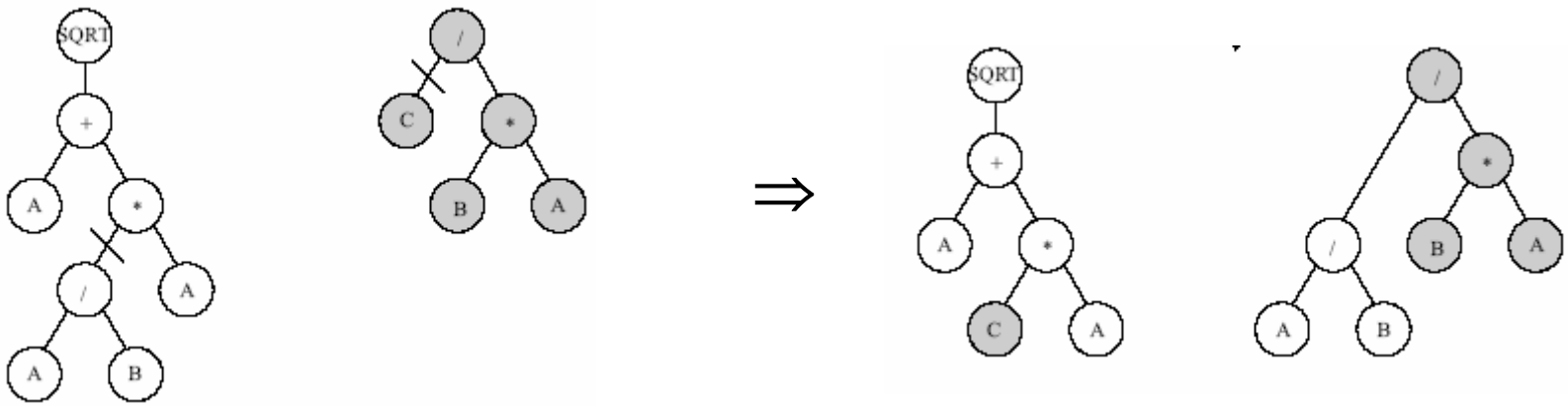
- Nella codifica binaria classica, il numero 0111=7 non è adiacente al numero 1000=8 , nonostante i numeri 7 e 8 differiscano di un'unità. In tal modo se 8 è una soluzione migliore di 7, l'algoritmo genetico deve cambiare tutti i bit per ottenerlo.
- La codifica grigia risolve il problema perché interi vicini sono rappresentati da stringhe che differiscono di un solo bit.

```
decodifica grigia (genotipo g){
   int n1=0;
   int x=0
   per i da 0 a l-1
        Se(g[i]=1){
                n1 = n1 +1
                x = x + (n1 mod 2)*2^(l-1-i)
        }
   restituisci x
}
```

Valore intero	Codifica binaria	Codifica grigia
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

Codifica ad albero (Programmazione Genetica)

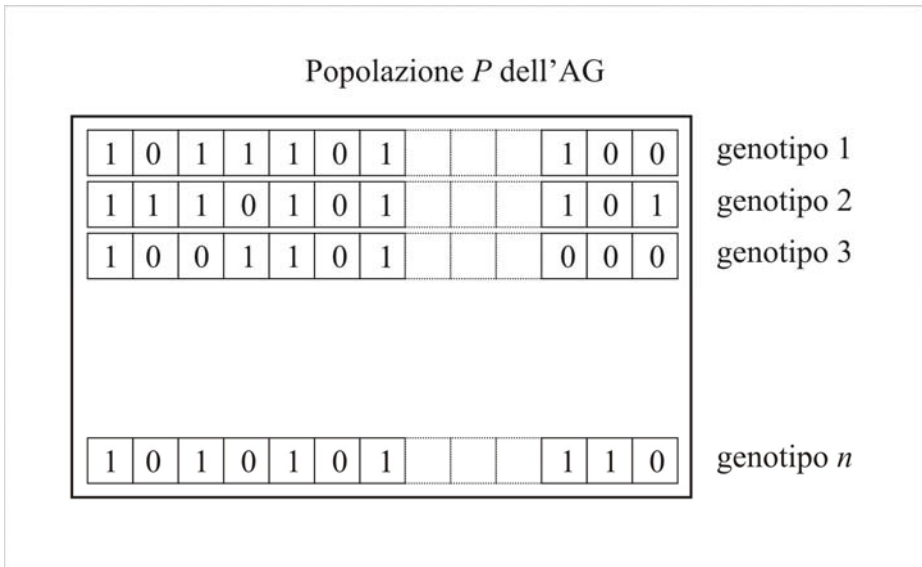
- La codifica ad albero è utilizzata nell'evoluzione di programmi, dove, piuttosto che generare direttamente soluzioni di un dato problema di ottimizzazione, l'obbiettivo è quello di derivare algoritmi in grado di risolvere particolari problemi computazionali.
- La struttura dati utilizzata è un albero con nodi terminali, detti foglie, e nodi non terminali. I nodi terminali, dai quali discendono sottoalberi, possono essere costanti e variabili specifiche del problema, mentre i nodi non terminali possono essere funzioni come **+, -, *, /** , √ e strutture di controllo del tipo **if then else**.



- L'operatore di crossover e di mutazione presentano alcune differenze rispetto ai corrispondenti operatori degli AG.
- L'incrocio consiste nella scelta di due punti di taglio, uno per ogni genitore, e nello scambio dei sottoalberi discendenti.
- L'operatore di mutazione, quando viene applicato, sostituisce sottoalberi con nuovi sottoalberi generati casualmente. Anche in questo caso le dimensioni dei programmi possono aumentare o diminuire.

L'originario modello di Holland

- L'originario modello di Holland opera su su una popolazione P di n stringhe di bit (dette individui o genotipi) di lunghezza l fissata



Funzione di fitness, spazio di ricerca e paesaggio d'idoneità

- La funzione di fitness valuta la bontà degli individui g_i della popolazione P nel risolvere il problema di ricerca dato:

$$f : P \rightarrow (-\infty, +\infty); \quad f(g_i) = f_i$$

- L'insieme delle stringhe binarie di lunghezza l ha 2^l elementi; tale insieme rappresenta lo spazio di ricerca (search space) dell'AG, cioè lo spazio che l'AG deve esplorare per risolvere il problema di ricerca (es. trovare il massimo o il minimo).

- I valori di fitness sui punti dello spazio di ricerca è detto paesaggio d'idoneità (fitness landscape).

Esempio di paesaggio d'idoneità di un AG con genotipi binari di 2 bit

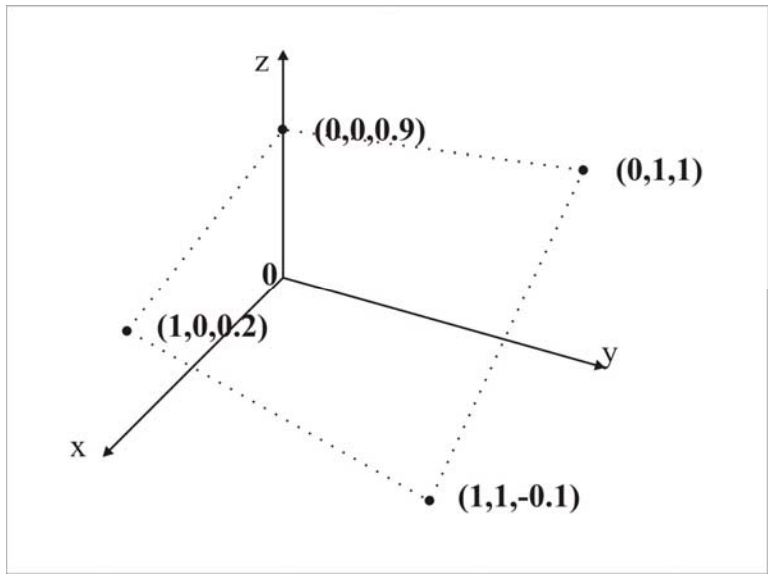
- Il numero di stringhe binarie di lunghezza 2 è

$$2^l = 2^2 = 4$$

- Lo spazio di ricerca dell'AG è dunque

$$S = \{(0,0), (0,1), (1,0), (1,1)\}$$

- I valori di fitness sui punti di S definiscono il paesaggio d'idoneità dell'AG



Operatori

- Una volta che la funzione di fitness ha determinato il valore di bontà di ogni individuo della popolazione, una nuova popolazione di individui (o genotipi) viene creata applicando alcuni operatori che si ispirano alla selezione naturale e alla genetica

- Gli operatori proposti da Holland sono:
  - Selezione (ispirato alla selezione naturale)
  - Crossover (ispirato alla genetica)
  - Mutazione (ispirato alla genetica)

- Gli ultimi due sono detti operatori genetici

L'operatore di selezione

- La selezione naturale Darwiniana sostiene che gli individui più "forti" abbiano maggiori probabilità di sopravvivere nell'ambiente in cui vivono e, dunque, maggiore probabilità di riprodursi

- Nel contesto dell'AG di Holland, gli individui più forti sono quelli con fitness più alta, poiché risolvono meglio di altri il problema di ricerca dato; per questo essi devono essere privilegiati nella fase di selezione di quegli individui che potranno riprodursi dando luogo a nuovi individui

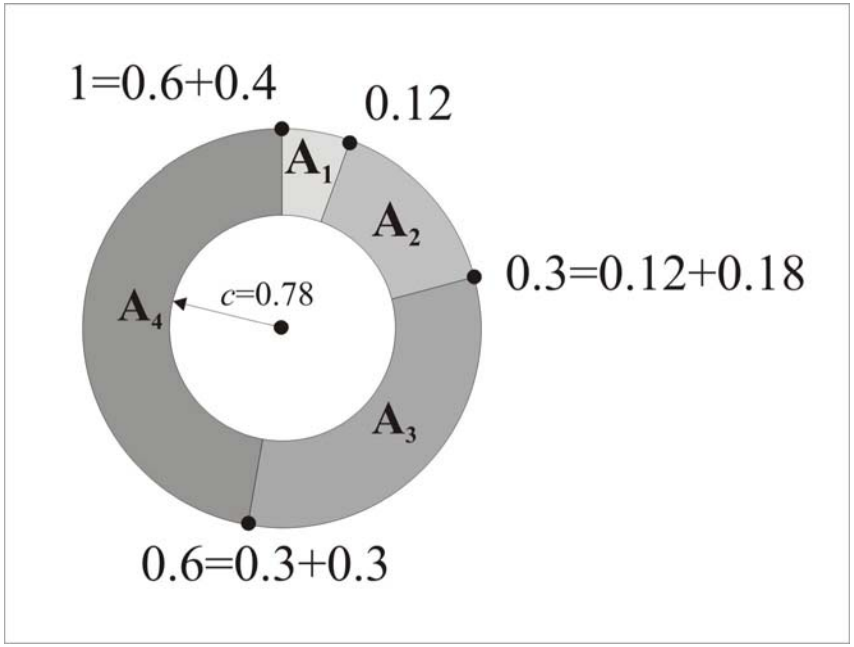
La selezione proporzionale

- Holland propose un metodo di selezione proporzionale al valore di fitness; sia f_i il valore di fitness del genotipo g_i, la probabilità che g_i sia selezionato per la riproduzione è:

$$p_{s,i} = f_i / \Sigma f_j$$

- Tali probabilità sono utilizzate per costruire una sorta di roulette

Esempio di roulette



- I quattro individui A_1, A_2, A_3 e A_4, con probabilità di selezione 0.12, 0.18, 0.3 e 0.4, occupano uno "spicchio" di roulette di ampiezza pari alla propria probabilità di selezione. Nell'esempio l'operatore di selezione genera il numero casuale $c=0.78$ e l'individuo A_4 viene selezionato

Mating pool

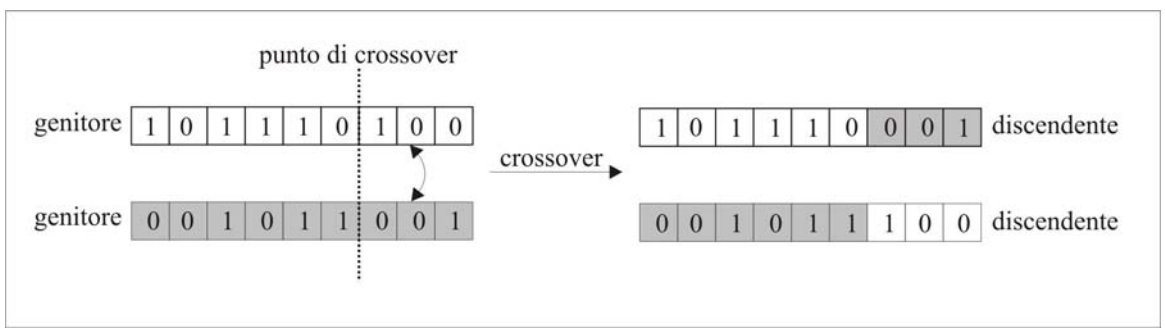
- Ogni volta che un individuo della popolazione è selezionato ne viene creata una copia; tale copia è inserita nel così detto mating pool (piscina d'accoppiamento)

- Quando il mating pool è riempito con esattamente n (numero di individui della popolazione) copie di individui della popolazione, nuovi n discendenti sono creati applicando gli operatori genetici

Crossover

- Si scelgono a caso due individui nel mating pool (genitori) e un punto di taglio (punto di crossover) su di essi. Le porzioni di genotipo alla destra del punto di crossover vengono scambiate generando due discendenti.

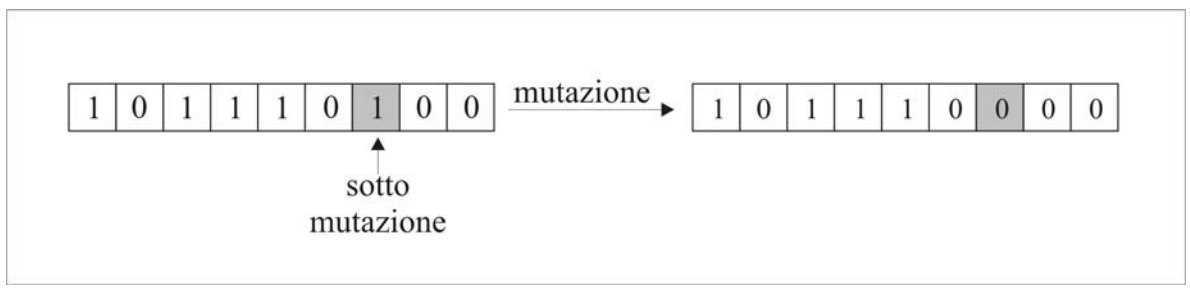
- L'operatore di crossover è applicato, in accordo a una prefissata probabilità p_c, $n/2$ volte in modo da ottenere n discendenti; nel caso in cui il crossover non sia applicato, i discendenti coincidono con i genitori.



Mutazione

- Una volta che due discendenti sono stati generati per mezzo del crossover, in funzione di una pressata e usualmente piccola probabilita p_m, il valore dei bit dei nuovi individui sono cambiati da 0 in 1 o viceversa.

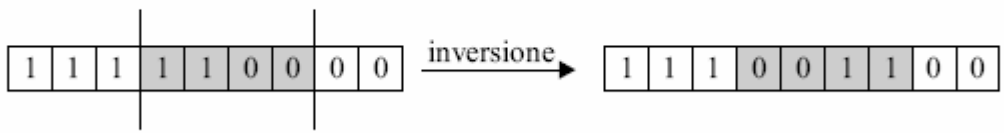
- Come il crossover rappresenta una metafora della riproduzione sessuale, l'operatore di mutazione modella il fenomeno genetico della rara variazione di elementi del genotipo negli esseri viventi durante la riproduzione sessuale.



Inversione

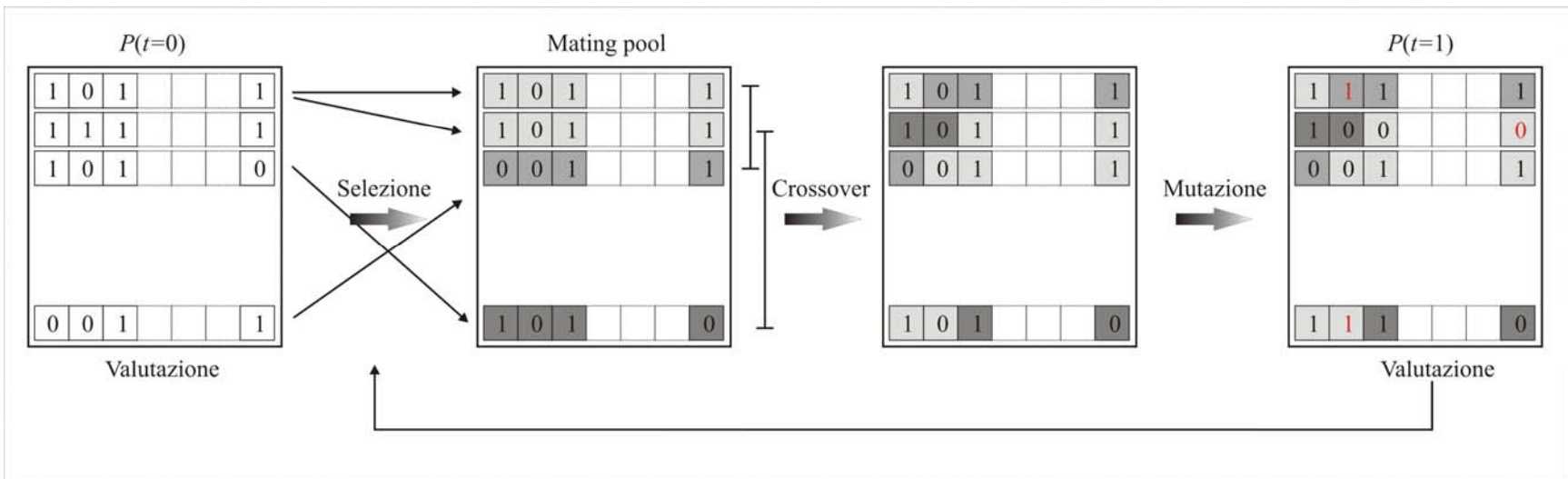
- l'operatore di **inversione**, ispirato come i precedenti a un analogo meccanismo biologico, in accordo a una prefissata probabilità $p_{inversion}$, sceglie casualmente due punti nella stringa che codifica l'individuo e inverte i bit tra le due posizioni.

- In biologia il significato di un gene spesso non dipende dalla sua posizione nel cromosoma e, di conseguenza, l'inversione ne lascia invariata la semantica. Purtroppo questo non è vero per la maggior parte degli algoritmi genetici e l'uso di tale operatore porta inevitabilmente a complicazioni a volte anche molto significative. Per tale ragione l'inversione è raramente utilizzata nelle applicazioni pratiche.



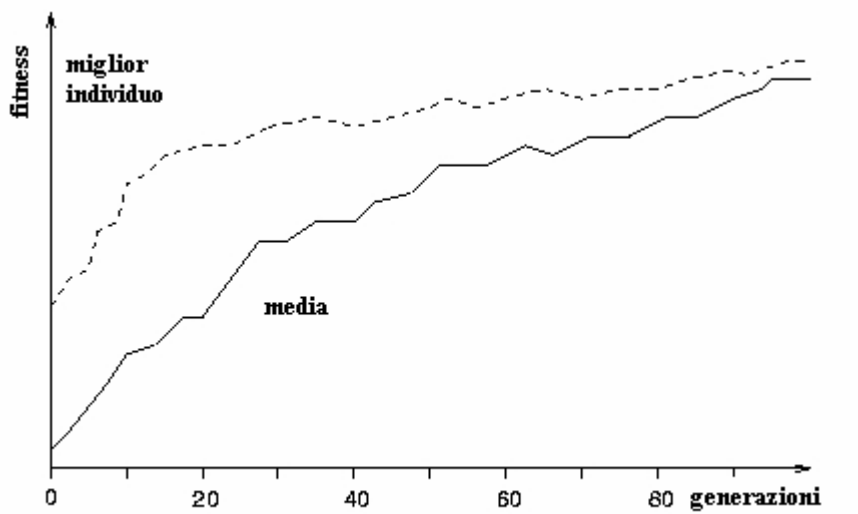
Schema iterativo

```
AG {
    t=0
    inizializza la popolazione P(t) in maniera casuale
    valuta la popolazione P(t)
    mentre (!criterio_fermata) {
        t=t+1
        crea P(t) applicando selezione, crossover e mutazione
        valuta P(t)
    }
}
```



Convergenza

- Se l'AG è correttamente implementato, la popolazione evolverà in molte generazioni in modo che il valore di fitness del miglior individuo e la media in ogni generazione cresca verso l'ottimo globale.
- La convergenza è la progressione verso la crescente uniformità: si può definire che un gene converga quando il 95% della popolazione condivide lo stesso valore.
- **La popolazione converge quando tutti i geni convergono.**



Esempio

- **Trovare il massimo della funzione $y=x$ nell'intervallo [0,255]**

1. Scrivere un programma che implementi l'AG oppure trovarne una su Internet!
2. Scegliere la dimensione della popolazione
3. Scegliere la lunghezza del genotipo
4. Scegliere una funzione di fitness
5. Scegliere una probabilità di crossover
6. Scegliere una probabilità di mutazione

Schemi e parallelismo implicito

Definizione 4.1. Sia $V = \{0, 1, \ldots, k\}$ l'alfabeto di k simboli su cui sono costruite le stringhe, di lunghezza l fissata, dell'algoritmo genetico. Si definisce schema una stringa di lunghezza l costruita sull'alfabeto esteso $V+ = \{0, 1, \ldots, k, *\}$.

Osservazione 4.1. Il numero di stringhe che si possono costruire su V è k^l, mentre il numero di schemi è $(k+1)^l$. Per esempio se $k = 2$ ed $l = 5$, si avranno $k^l = 2^5 = 32$ individui e $(k+1)^l = 3^5 = 243$ schemi.

Uno schema rappresenta un insieme di stringhe che hanno un qualsiasi simbolo di V nelle corrispondenti posizioni in cui compare il simbolo $*$ o, equivalentemente, rappresenta un insieme di stringhe con caratteristiche comuni nelle corrispondenti posizioni in cui non compare il simbolo $*$.

Esempio 4.1. Si considerino $l = 5$ e $V = \{0, 1\}$. Allora:

- lo schema $H_1 = *0000$ rappresenta l'insieme costituito dalle due stringhe aventi uno dei due simboli di V nella posizione 0 e il sibolo 0 nelle posizioni 2, 3, 4 e 5, cioè
$$H_1 = \{00000, 10000\}$$
- lo schema $H_2 = *111*$ rappresenta l'insieme costituito dalle quattro stringhe aventi uno dei due simboli di V nelle posizioni 1 e 5 e il simbolo 1 nelle posizioni 2, 3 e 4, cioè
$$H_2 = \{01110, 01111, 11110, 11111\}$$

Schemi e parallelismo implicito

- lo schema $H_3 = 0*1**$ rappresenta l'insieme costituito dalle otto stringhe aventi uno dei due simboli di V nelle posizioni 2, 4 e 3, uno 0 nella posizione 1 e un 1 nella posizione 3, cioè

$$H_3 = \{00100, 00101, 00110, 00111, 01100, 01101, 01110, 01111\}$$

Definizione 4.2. Si definisce cardinalità di uno schema H, e si indica con $\#H$, il numero di stringhe dello schema, dato dalla cardinalità dell'alfabeto su cui sono costruiti gli individui dell'AG elevato al numero di simboli $*$ presenti nello schema.

Esempio 4.2. Nell' esempio precedente, poichè $\#V = 2$, risulta: $\#H_1 = 2^1 = 2$, $\#H_2 = 2^2 = 4$ e $\#H_3 = 2^3 = 8$.

Senza perdere di generalità, consideriamo da ora in poi l'alfabeto binario $V = \{0, 1\}$.

Teorema 4.1. *Il numero di schemi di un algoritmo genetico presenti in una popolazione di n individui di lunghezza l costruiti sull'alfabeto* $V = \{0, 1\}$ *è compreso tra* 2^l *e* $n2^l$.

Dimostrazione. Consideriamo una generica stringa $A = a_1 a_2 \dots a_l$ tale che $a_i \in V = \{0, 1\} \quad \forall i \in \{1, 2, \dots, l\}$. Essa è un membro di ogni schema che abbia nelle corrispondenti posizioni di A o lo stesso simbolo di A oppure il simbolo $*$. Per esempio A è membro dello schema $H = *a_2 * * \dots * a_l$. Dunque, ogni schema che contenga A è vincolato ad avere in ogni posizione uno tra due simboli: il corrispondente simbolo di A, oppure $*$. Il loro numero è, pertanto, 2^l. Può succedere che, per esempio quando tutte le stringhe della popolazione sono ugluali, tutti gli individui siano membri degli stessi schemi; in tal caso il numero di schemi della popolazione è 2^l. All'opposto, quando tutte le stringhe della popolazione sono membri di schemi diversi, il loro numero è $n2^l$. □

Dunque, a ogni generazione, l'algoritmo genetico processa esplicitamente n individui e, allo stesso tempo, un numero molto maggiore di schemi implicitamente. Tale caratteristica è chiamata parallelismo implicito.

Il teorema fondamentale degli Algoritmi Genetici

Definizione 4.3. Si definisce ordine di uno schema H, e si indica con $o(H)$, il numero di posizioni fisse presenti nel modello.

Esempio 4.3.

$$o(011*1**) = 4$$

$$o(0******) = 1$$

Definizione 4.4. Si definisce lunghezza caratteristica di uno schema H, e si indica con $\delta(H)$, la distanza tra la prima e l'ultima posizione fissa dello schema.

Esempio 4.4.

$$\delta(h_1h_2h_3h_4h_5h_6h_7) = \delta(011*1**) = 5 - 1 = 4$$

$$\delta(h_1h_2h_3h_4h_5h_6h_7) = \delta(0******) = 1 - 1 = 0$$

Consideriamo ora gli effetti della selezione, del crossover e della mutazione sul numero di rappresentanti degli schemi dell'algoritmo genetico. Supponiamo che a una data generazione t esistano $m = m(H,t)$ rappresentanti di un particolare schema H nella popolazione P(t). Come descritto nel paragrafo 3, formula 1, ogni individuo, quindi anche ogni rappresentante di H, è selezionato per la riproduzione con probabilità $p_{selection,i} = \frac{f_i}{\sum_{j=1}^{n} f_j}$. Dunque, la probabilità che una stringa dello schema venga selezionata a ogni tentativo è

$$\sum_{i=1}^{m} p_{selection,i} = \frac{\sum_{i=1}^{m} f_i}{\sum_{j=1}^{n} f_j} \tag{2}$$

Poichè l'operatore di selezione è eseguito n volte, il numero atteso di rappresentanti di H è

$$m(H,t+1) = n \sum_{i=1}^{m} p_{selection,i} = n \frac{\sum_{i=1}^{m} f_i}{\sum_{j=1}^{n} f_j} = \frac{\sum_{i=1}^{m} f_i}{\bar{f}} \tag{3}$$

dove $\bar{f}$ è la fitness media della popolazione $P(t)$. Moltiplicando ambo i membri per $m(H,t)$ si ha

$$m(H,t+1) = \frac{m(H,t)}{m(H,t)} \frac{\sum_{i=1}^{m} f_i}{\bar{f}} = m(H,t) \frac{f(H)}{\bar{f}} \tag{4}$$

dove $f(H)$ è la fitness media degli individui rappresentanti dello schema H. La formula 4 indica che se la fitness media dei rappresentanti di uno schema H è maggiore della fitness media della popolazione, allora il numero di rappresentanti dello schema aumenta, altrimenti diminuisce. Per determinare la velocità con cui varia il numero di

aumenta, altrimenti diminuisce. Per determinare la velocità con cui varia il numero di rappresentanti scriviamo

$$f(H) = \bar{f} + c\bar{f} \tag{5}$$

dove c è un costante reale, minore di uno se la fitness media dello schema è minore della fitness media della popolazione, maggiore di uno altrimenti. Allora

$$m(H, t+1) = m(H, t)\frac{\bar{f} + c\bar{f}}{\bar{f}} = m(H, t)(1 + c) \tag{6}$$

per cui si avrà

$$m(H, t) = m(H, t-1)(1 + c) = m(H, t-2)(1 + c)^2 = \ldots$$

$$\ldots = m(H, 0)(1 + c)^t \tag{7}$$

Dunque, il numero di rappresentanti di uno schema cresce, o decresce, esponenzialmente durante le generazioni dell'algoritmo genetico a seconda che la fitness media dello schema sia maggiore o minore della fitness media della popolazione

Consideriamo ora un individio A e due schemi, H_1 e H_2, di cui A sia un rappresentante

$$A = 0111000$$

$$H_1 = *1****0$$

$$H_2 = ***10**$$

e supponiamo che A sia sogetto a crossover con punto di taglio nella posizione 3, cioè dopo il terzo gene

$$A = 011 \mid 1000$$

$$H_1 = *1* \mid ***0$$

$$H_2 = *** \mid 10**$$

Osserviamo che i discendenti di A non saranno rappresentanti dello schema H_1, a meno che A non si incroci con un individuo che abbia anch'esso un 1 nella posizione 2 e uno zero nella posizione 7. Al contrario, almeno un discendente di A sarà ancora rappresentante di H_2. Lo schema H_1 ha minori probabilità di sopravvivere al crossover rispetto ad H_2 perchè ha maggiori probabilità che il punto di taglio cada tra le due posizioni fisse estreme. Infatti $\delta(H_1) = 7 - 2 = 5$, e il punto di taglio può cadere con eguale probabilità in uno degli $l - 1 = 7 - 1 = 6$ posizioni. Allora le probabilità che H_1 e H_2 vengano distrutti sono

$$p_d(H_1) \leq \frac{\delta(H_1)}{l-1} = \frac{5}{6}$$

$$p_d(H_2) \leq \frac{\delta(H_2)}{l-1} = \frac{1}{6}$$

Pertanto, definiamo la probabilità che uno scema H venga distrutto come

$$p_d(H) \leq \frac{\delta(H)}{l-1} \tag{8}$$

Se il crossover è eseguito con probabilità $p_{crossover}$, la formula precedente diventa

$$p_d(H) \leq p_{crossover}\frac{\delta(H)}{l-1} \tag{9}$$

La probabilità di sopravvivenza di uno schema H sarà, dunque

$$P_s \geq 1 - p_d = 1 - p_{crossover}\frac{\delta(H)}{l-1} \tag{10}$$

L'espressione precedente afferma che gli schemi che anno alte probabilità di sopravvivere al crossover sono quelle che hanno lunghezze caratteristiche piccole. Moltiplicando la 4 con la 10 otteniamo il numero atteso di rappresentanti dello schema H per effetto della selezione e del crossover:

$$m(H, t+1) >= m(H, t)\frac{f(H)}{\bar{f}}(1 - p_{crossover}\frac{\delta(H)}{l-1}) \tag{11}$$

Affinchè uno schema H sopravviva anche all'operatore di mutazione, tutte le posizioni fisse $o(H)$ devono rimanere immutate. La probabilità di sopravvivenza sarà, dunque:

$$p'_s = (1 - p_{mutation})^{o(H)} \tag{12}$$

se $p_{mutation} \ll 1$, allora

$$p'_s = (1 - p_{mutation})^{o(H)} \approx 1 - o(H)p_{mutation} \tag{13}$$

Moltiplicando la 11 per la 13 otteniamo il numero atteso di rappresentanti dello schema H per effetto della selezione del crossover e della mutazione:

$$m(H, t+1) >= m(H, t)\frac{f(H)}{\bar{f}}(1 - p_{crossover}\frac{\delta(H)}{l-1})(1 - o(H)p_{mutation}) =$$

$$= m(H, t)\frac{f(H)}{\bar{f}}(1 - o(H)p_{mutation} - p_{crossover}\frac{\delta(H)}{l-1} +$$

$$+ p_{crossover}\frac{\delta(H)}{l-1}o(H)p_{mutation}) \tag{14}$$

Il termine $p_{crossover}\frac{\delta(H)}{l-1}o(H)p_{mutation}$ si può trascurare perchè abbastanza piccolo, per cui la formula precedente diventa

$$m(H, t+1) >= m(H, t)\frac{f(H)}{\bar{f}}(1 - p_{crossover}\frac{\delta(H)}{l-1} - o(H)p_{mutation}) \tag{15}$$

Il risultato precedente prende il nome di *teorema fondamentale degli algoritmi genetici*: il numero atteso di rappresentanti di uno schema H, con fitness media maggiore della fitness media della popolazione e lunghezza caratteristica piccola, aumenta esponenzialmente (6) di generazione in generazione.

In conclusione, l'algoritmo genetico ha la capacità di esplorare un numero compreso tra 2^l e $n2^l$ schemi per ogni generazione (parallelismo implicito) e di concentrare il maggior numero di individui nelle zone dello spazio di ricerca più promettenti poichè premia gli schemi caratterizzati da fitness alta (teorema fondamentale degli algoritmi genetici).

Vari tipi di selezione: terminologia

- Gli operatori di selezione possono rimpiazzare l'intera popolazione (**AG generazionali**) o soltanto una parte di essa (**AG steady state** ovvero a stato stazionario).
- Inoltre, gli operatori di selezione possono scegliere piu volte o soltanto una volta uno stesso individuo per la riproduzione.
  - Nel primo caso si parla di **operatori con rimpiazzamento** (with replacement), nel senso che l'individuo selezionato per la riproduzione, dopo l'accoppiamento, viene reinserito nella vecchia popolazione e puo essere selezionato nuovamente.
  - Nel secondo caso si parla di **operatori senza rimpiazzamento** (without replacement).
- Sia negli AG generazionali che negli AG steady-state può succedere che l'individuo migliore venga perso nel passaggio alla generazione successiva.
- I modelli che garantiscono la sopravvivenza del miglior individuo sono detti **elitistici**, o ***k*-elitistici** se garantiscono la sopravvivenza dei migliori ***k*** individui.

Considerazioni sull'operatore di selezione

- La selezione incide in maniera significativa sulla dinamica dell'algoritmo genetico:
  - una pressione selettiva troppo forte può dar luogo alla predominanza di qualche individuo con fitness particolarmente alta e condurre l'algoritmo genetico in un ottimo locale da cui difficilmente riuscirà a uscire;
  - d'altro canto, una pressione selettiva troppo debole può dar luogo a un eccessivo aumento del tempo d'esecuzione necessario a trovare una soluzione accettabile.
- L'originario modello di Holland impiega la selezione proporzionale alla fitness utilizzando uno schema con rimpiazzamento.
  - Tale schema presenta una pressione selettiva molto alta: gli individui con fitness alta e i loro discendenti si moltiplicano troppo velocemente e impediscono di fatto all'algoritmo genetico di effettuare ulteriori significative esplorazioni.

Selezione a torneo

- La selezione a torneo non richiede operazioni globali sulla popolazione e risulta più efficiente in termini di tempo di calcolo.
- Nel tipo più comune di selezione a torneo, si scelgono a caso due individui e si genera un numero casuale $c \in [0,1]$.
- Se c risulta minore di un parametro $k \in [0,1]$, (per esempio k=0,75), allora si seleziona il piu idoneo, altrimenti si sceglie il meno idoneo.
- Se si applica uno schema con rimpiazzamento, inoltre, i due individui sono reinseriti nella vecchia popolazione e possono essere scelti di nuovo.

Selezione di Boltzmann

- La selezione di Boltzmann permette una pressione selettiva variabile durante l'evoluzione dell'algoritmo genetico.
- Inizialmente una pressione selettiva bassa consente agli individui meno idonei di riprodursi quasi quanto quelli più idonei. Questo permette di mantenere una significativa diversità nella popolazione e un buon campionamento dello spazio delle soluzioni. Successivamente, l'aumento della pressione selettiva durante l'evoluzione dell'algoritmo genetico favorisce la riproduzione degli individui piu idonei.
- Se nella fase iniziale, caratterizzata da bassa pressione selettiva, l'algoritmo genetico riesce a individuare la zona giusta dello spazio di ricerca, l'aumento della pressione selettiva guida la ricerca proprio in quella zona favorendo la convergenza verso una buona soluzione.
- Una tipica applicazione della selezione di Boltzmann consiste nell'assegnare a ogni individuo g_i un valore

$$ExpVal(g_i, t) = \frac{e^{f_i/T}}{\langle e^{f_i/T} \rangle_t}$$

- Dove $\langle\rangle_t$ indica la media nella generazione t, T è la variabile "temperatura" che, diminuendo gradualmente all'aumentare di t garantisce l'aumento della pressione selettiva.

Selezione in base al rango

- La selezione in base al rango riduce la pressione selettiva rispetto alla selezione proporzionale. Gli individui della popolazione sono classificati in base alla fitness e i relativi valori attesi dipendono dalla posizione (rango) che occupano nella classifica.
- Nel metodo di classificazione lineare ogni individuo è classificato in ordine crescente di idoneità, da 1 a ***n***.
- ***Max*≥*0*** è il valore dell'individuo di rango ***n***, ***Min*** è il valore dell'individuo di rango **1**.
- Il valore di ogni individuo g_i della popolazione alla generazione ***t*** è dato da

$$ExpVal(g_i, t) = Min + (Max - Min)\frac{rango(g_i, t) - 1}{n - 1}$$

- affinché la consistenza numerica della popolazione si mantenga inalterata tra una generazione e l'altra, è necessario che **1≤*Max*≤2** e ***Min*=2-*Max***

Riferimenti bibliografici

[1] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. Wiley, 1966.

[2] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[3] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, second edition: mit press, 1992 edition, 1975.

[4] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1996.

PGAPack

- PGAPack è una libreria “free” per AG scaricabile dal sito

http://www-fp.mcs.anl.gov/CCST/research/reports_pre1998/comp_bio/stalk/pgapack.html

- Implementa l’AG di Holland e numerosi modelli proposti successivamente (alcuni di essi saranno analizzati nella lezione successiva)
- Gira su ambienti operativi UNIX e UNIX-like (es. Linux)
- Implementa anche una versione parallela di AG che prende il nome di modello master slave. Dunque può sfruttare più unità di calcolo simultaneamente.