

Introduzione alla Computazione Evolutiva

**Pier Luca Lanzi
Dipartimento di Elettronica e Informazione
Politecnico di Milano
Venerdì 7 Marzo 1997**

Sommario

- Caratteristiche dei Modelli di Calcolo Evolutivo**
- Algoritmi Genetici**
- Strategie Evolutive**
- Genetic Programming**

Computazione Evolutiva

Raggruppa un vasto insieme di modelli di calcolo che si ispirano alla metafora della genetica delle popolazioni e alla teoria Darwiniana dell'evoluzione.

Caratteristiche:

Popolazione di individui che evolve nel tempo.

Il meccanismo di selezione naturale favorisce la sopravvivenza degli individui migliori.

Mutazioni avvengono di tanto in tanto in alcuni individui della specie.

Evoluzione Naturale - Evoluzione Artificiale

Evoluzione Naturale:

L'evoluzione non è diretta. Gli individui sono migliori se in generale sopravvivono e si adattano ai mutamenti che possono avvenire nell'ambiente.

Evoluzione Artificiale:

Si può paragonare all'allevamento. Ci sono mutamenti casuali nella popolazione ma è l'allevatore a decidere di volta in volta quali sono le caratteristiche interessanti che devono essere selezionate. L'evoluzione è in questo caso diretta.

Evoluzione Artificiale

- L'Evoluzione Naturale può essere modellata attraverso gli Algoritmi Evolutivi.
- In pratica gli algoritmi evolutivi modellano l'attività di un allevatore.
- Viene mantenuta una popolazione di individui.
- Avvengono mutazioni in maniera casuale nella popolazione.
- La selezione umana favorisce gli individui con le caratteristiche desiderate: gli individui con le caratteristiche migliori vengono utilizzati nella riproduzione.

Evoluzione - Terminologia

Evoluzione Naturale

Evoluzione Artificiale

Fitness

Funzione di Fitness/Obiettivo

Cromosoma/Individuo

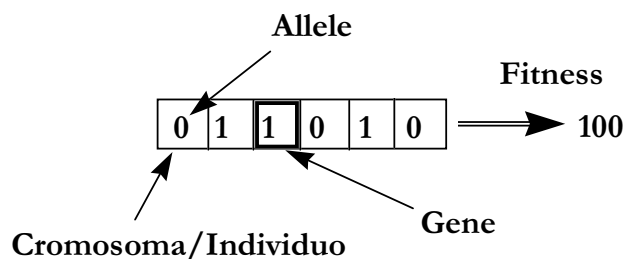
Stringa/Soluzione

Gene

Gene/Parametro/Feature

Allele

Allele/Valore del Parametro



Algoritmi Genetici (Holland 1975)

Popolazione

È costituita da un insieme di stringhe binarie di lunghezza finita. Viene definita una funzione di fitness che associa un valore ad ogni possibile stringa binaria.

Riproduzione/Selezione

Ogni individuo ha una probabilità di sopravvivenza proporzionale alla fitness.

Mutazione/Ricombinazione

Esistono due operatori: **CROSSOVER** e **MUTAZIONE** per la ricombinazione degli individui nella popolazione.

Algoritmi Genetici - Algoritmo

- $t := 0$;
- Inizializza la popolazione iniziale $P(t)$ con N individui generati in maniera casuale.
- **RIPETI**
 - Calcola la fitness per tutti gli elementi di $P(t)$;
 - Genera $P_r(t+1)$ da $P(t)$ applicando l'operatore di **RIPRODUZIONE**;
 - Genera $P_c(t+1)$ da $P_r(t+1)$ applicando l'operatore di **CROSSOVER**;
 - Genera $P_m(t+1)$ da $P_c(t+1)$ applicando l'operatore di **MUTAZIONE** a $P_c(t+1)$;
 - $P(t+1) := P_m(t+1)$;
 - $t := t+1$;
- **FINO** alla Condizione di Terminazione;

Algoritmi Genetici - Riproduzione/Selezione (1)

Individuo	Fitness	Probabilità di Riproduzione
x_1	$f(x_1)$	$P_r(x_i) = \frac{f(x_1)}{\sum_{i=1}^n f(x_i)}$
\vdots	\vdots	
x_j	$f(x_j)$	$\frac{f(x_j)}{\sum_{i=1}^n f(x_i)}$
\vdots	\vdots	
x_n	$f(x_n)$	$\frac{f(x_n)}{\sum_{i=1}^n f(x_i)}$

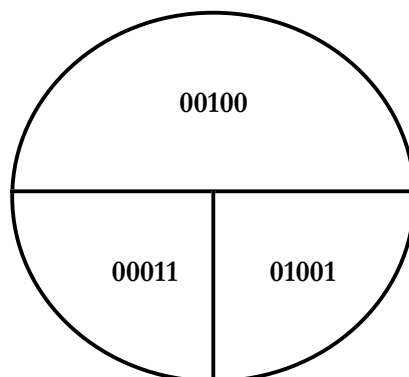
Ogni individuo ha una probabilità di riprodursi proporzionale a $P_r(x_i)$

Algoritmi Genetici - Riproduzione/Selezione (2)

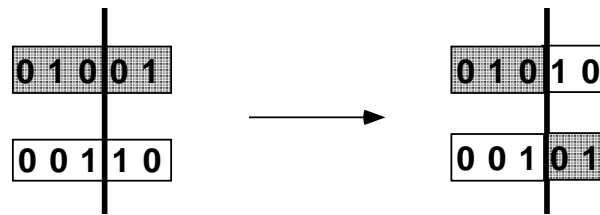
Esempio:

Individui	Fitness	Probabilità
00100	$f(00100) = 50$	$Pr(00100) = .5$
01001	$f(01001) = 25$	$Pr(01001) = .25$
00011	$f(00011) = 25$	$Pr(00011) = .25$

Selezione delle
offspring tramite
ROULETTE



Operatore di Ricombinazione: Crossover

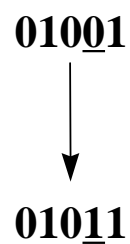


Vengono scelte $(N/2) \cdot p_c$ coppie di individui in maniera casuale dalla popolazione.

Un punto di taglio viene scelto in maniera casuale all'interno degli individui e il crossover è eseguito.

Operatore di Mutazione

Ogni allele di un individuo è mutato con probabilità P_m



Esempio - Ottimizzazione di Funzione (1)

Problema:

- Si vuole massimizzare la funzione $y=x^2$ sull'intervallo intero $[0..31]$
-

Passi:

- Scegliere una codifica per gli elementi dello spazio di ricerca;
- Scegliere una funzione di fitness;
- Applicazione dell'algoritmo genetico.

Esempio - Ottimizzazione di Funzione (2)

Codifica

Scegliamo la classica codifica binaria. Ci sono 32 valori possibili e quindi sono necessari 5bit.

Funzione di Fitness

La funzione di fitness è la stessa funzione da massimizzare $y=x^2$.

Esempio: $f(00100) = 4^2 = 16$

Popolazione Iniziale

$\{01101, 11000, 01000, 10011\}$

Esempio - Ottimizzazione di Funzione (3)

Selezione

N	Stringa	f(s)	% Rispetto al Totale
1	01101	169	14.4%
2	11000	576	49.2%
3	01000	64	5.5%
4	10011	361	30.9%

Totale Fitness: 1170

La roulette viene girata 4 volte e seleziona la nuova popolazione {01101, 11000, 11000, 10011}.

Esempio - Ottimizzazione di Funzione (4)

Crossover

Supponiamo di avere una Probabilità di Crossover pari a 0.5. $P_c = 0.5$.

Vengono scelte casualmente $(N/2) * P_c$ coppie di individui cioè $(4/2) * .5 = 1$ coppia di individui e il punto di crossover.

0110 1	\longrightarrow	0110 0
1100 0		1100 1

La Nuova Popolazione

{01100, 11001, 11000, 10011}

Esempio - Ottimizzazione di Funzione (5)

Mutazione

Con una probabilità di mutazione $P_m=0.01$ viene cambiato un il valore di un gene (allele) dell'individuo

10011



11011

Nuova Popolazione

{01100, 11001, 11000, 11011}

Strategie Evolutive

Comprendono una classe di algoritmi, ispirati dagli Algoritmi Genetici, in cui:

- Gli individui sono stringhe di elementi di un dominio qualunque e non più stringhe binarie.
 - Gli operatori genetici di Crossover e Mutazione sono ridefiniti per il nuovo dominio.
-

Esempi:

- Problemi di ottimizzazione: gli individui sono stringhe di variabili binarie e non sono più codificate.
- Problema del commesso viaggiatore: gli individui sono sequenze di nodi all'interno di un grafo.
- Genetic Programming: gli individui sono essere PROGRAMMI per computare una certa funzione.

Strategie Evolutive - Algoritmo (1)

L'algoritmo base per le strategie evolutive è praticamente lo stesso utilizzato per gli algoritmi genetici.

- $t := 0$;
- Genera una popolazione iniziale di μ individui: $P(0)$;
- RIPETI
 - Genera λ discendenti applicando gli operatori di mutazione e ricombinazione alla popolazione di μ individui. ($\lambda \geq \mu$)
 - Seleziona m individui che formeranno la nuova popolazione a partire dalla popolazione corrente e dai discendenti.
- FINO alla Condizione di Terminazione

Esempio - Problema del Commesso Viaggiatore

Otto città collegate. Dobbiamo trovare il percorso minimo che passa per tutte le città.

Primo Passo: Scelta di una rappresentazione per gli individui.

Un individuo codifica una sequenza di città.

a b c d e f g h

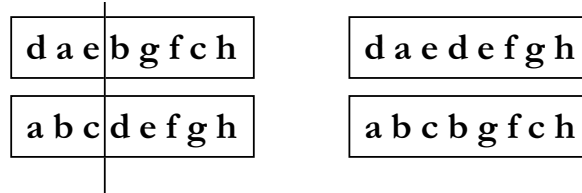
Secondo Passo: Scelta di una funzione di fitness.

$$\text{Fitness}(\text{Percorso}) = (\text{Lunghezza Percorso})^{-1}$$

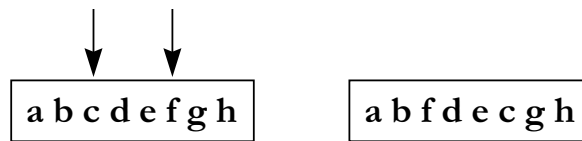
Esempio - Problema del Commesso Viaggiatore

Terzo: Definizione degli operatori di Crossover e Mutazione

Crossover Standard



Crossover Ridefinito



Esempio - Problema del Commesso Viaggiatore

Quarto: Scelta di un criterio di stop

- Numero fisso di passi di evoluzione.
- Si ferma l'evoluzione quando per un certo periodo non ci sono miglioramenti nella popolazione
- Quando l'errore stimato sulla soluzione è sufficientemente basso.

Computazione Evolutiva e Apprendimento Automatico

Genetic Programming

Algoritmi genetici che apprendono
programmi per risolvere problemi

Sistemi a Classificatori

Trovano insiemi di regole per un
sistema esperto allo scopo di
controllare agenti autonomi.

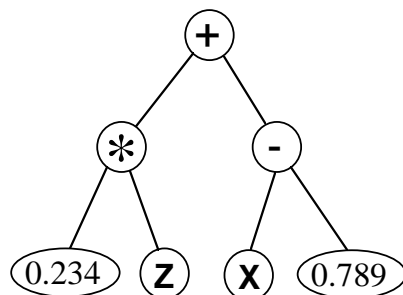
Genetic Programming (1)

La Genetic Programming (GP) è un algoritmo
evolutivo applicato a programmi LISP.

Gli individui sono S-Espressioni LISP.

I geni sono: *simboli terminali* o *funzioni primitive*.

Esempio: $T = \{X, Z, 0.234, 0.789\}$ $F = \{+, -, *\}$



$0.234 \cdot Z + (X - 0.789)$

S-espressione:

$(+ (* 0.234 Z) (- X 0.789))$