

Introduzione alla Computazione Evolutiva

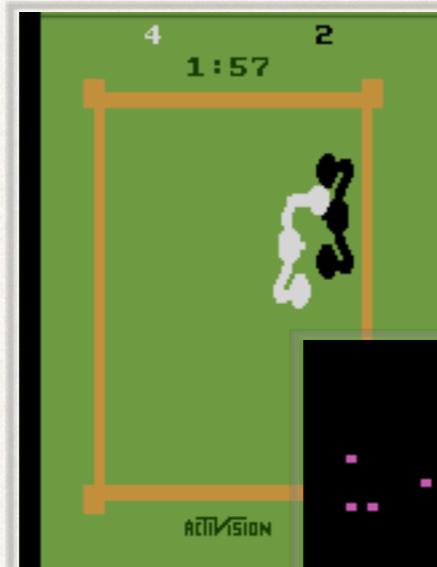
Luca Manzoni

3 Settembre 2020

Applicazioni degli algoritmi evolutivi



*The evolved antenna
of the NASA ST5 spacecraft (2006)*



*Genetic Programming: Evolution of Mona Lisa
R. Johansson (2008)*

*Evolving simple programs for playing Atari games
D.G. Wilson, S. Cussat-Blanc, H. Luga, J.F. Miller (2018)*

Problemi di ottimizzazione

- ✿ Molteplici soluzioni accettabili
- ✿ Ma alcune soluzioni sono meglio di altre
- ✿ Possiamo quantificare la qualità delle soluzioni
- ✿ Ma non è fattibile testare tutte le soluzioni

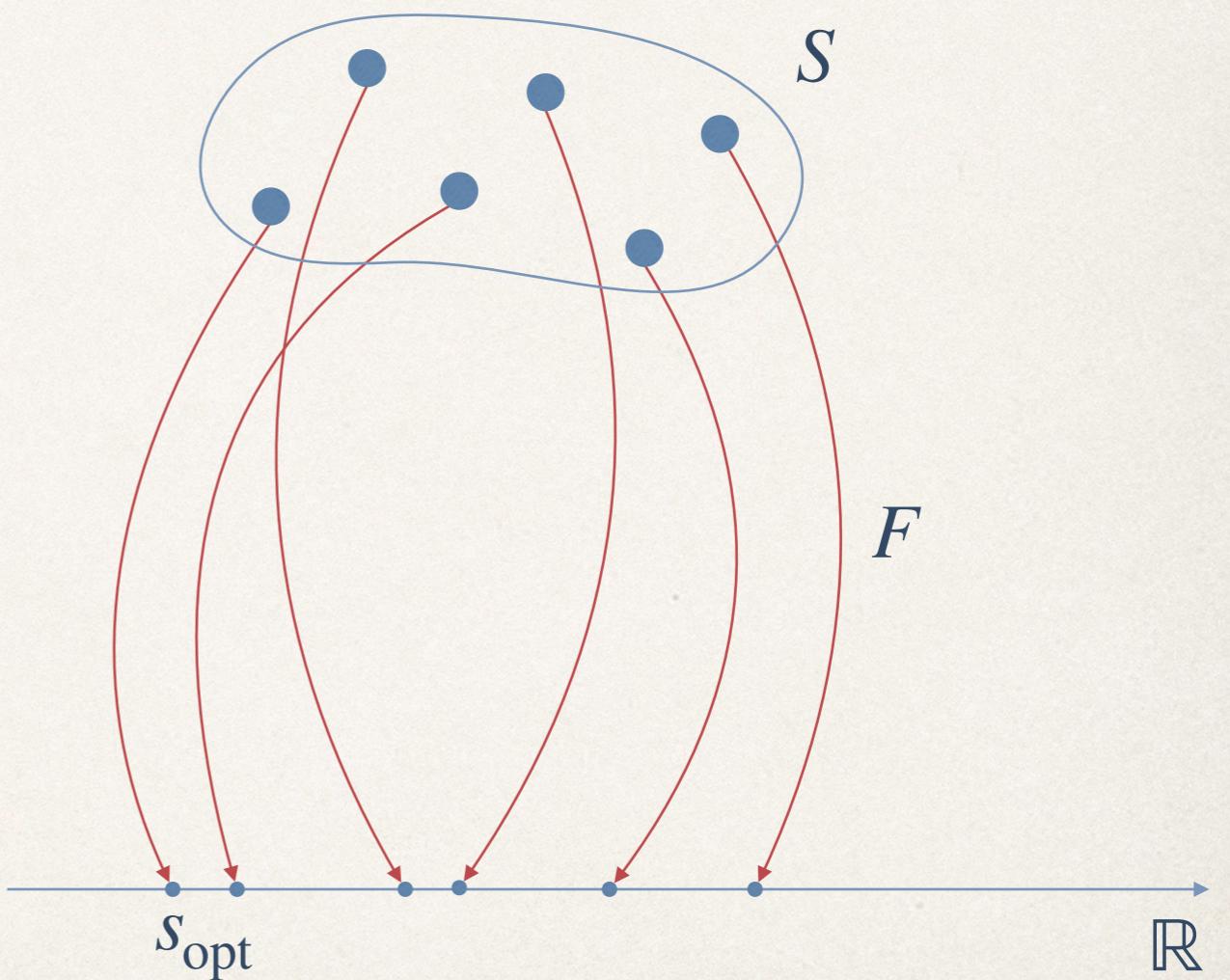
Problemi di ottimizzazione

Spazio delle soluzioni S

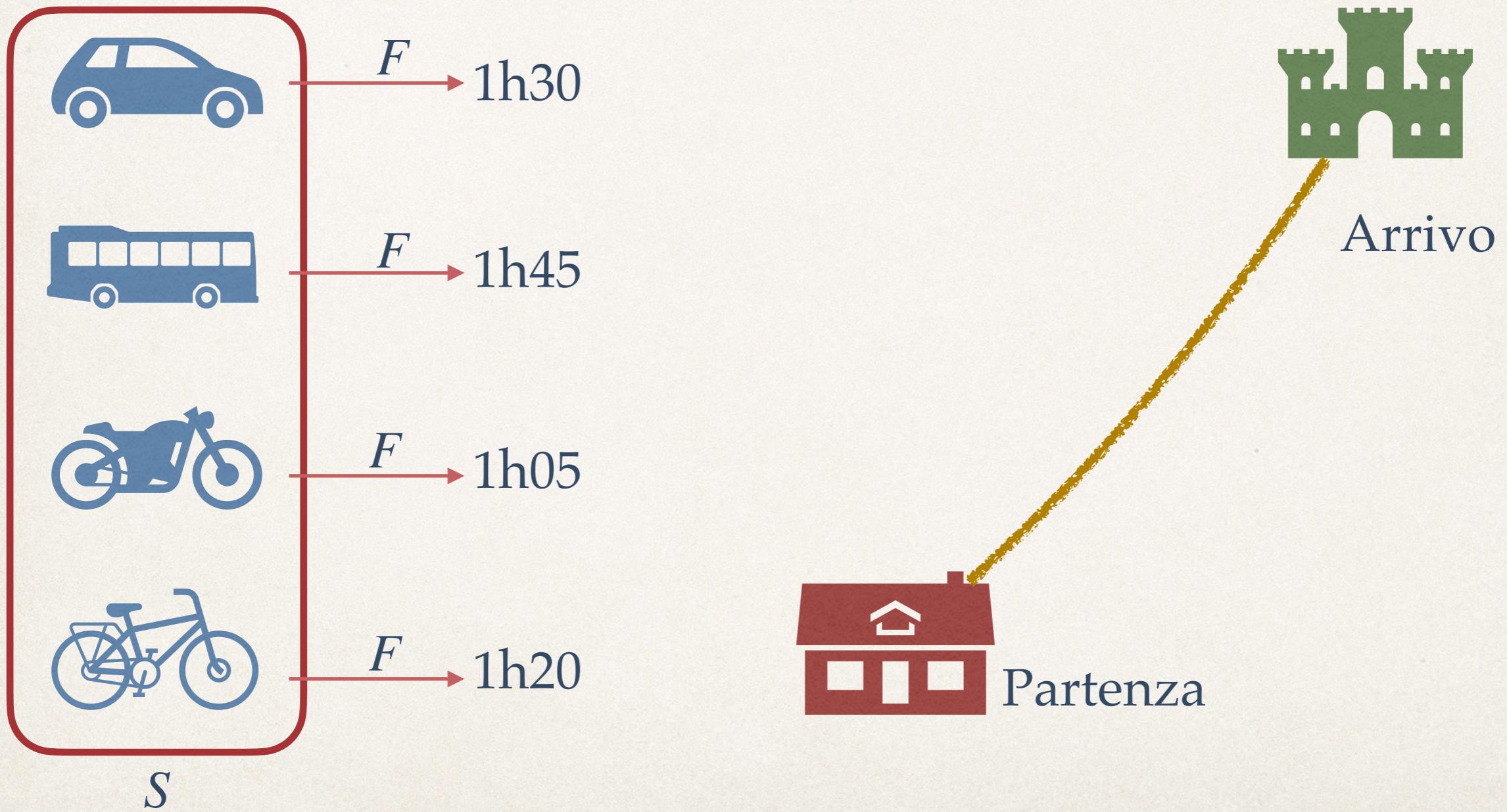
Funzione di fitness $F : S \rightarrow \mathbb{R}$

Soluzione ottima

$$s_{\text{opt}} = \underset{s \in S}{\operatorname{argmin}} F(s)$$



Problemi di ottimizzazione



Come Trovare l'Ottimo?

- ❖ La ricerca esaustiva è impossibile
- ❖ Non sono conosciuti algoritmi esatti efficienti
- ❖ E in alcuni casi costruire un algoritmo esatto potrebbe non essere possibile
- ❖ Quindi si usano delle euristiche

Ispirazione dalla Natura

Prendiamo esempio da un processo di ottimizzazione esistente

Qualcosa che è iniziato miliardi di anni fa

L'evoluzione naturale



Algoritmo Evolutivo di Base

- ❖ Selezione naturale

- ❖ Competizione per risorse limitate
- ❖ “Sopravvivenza del più forte”

Fenotipo

- ❖ Riproduzione

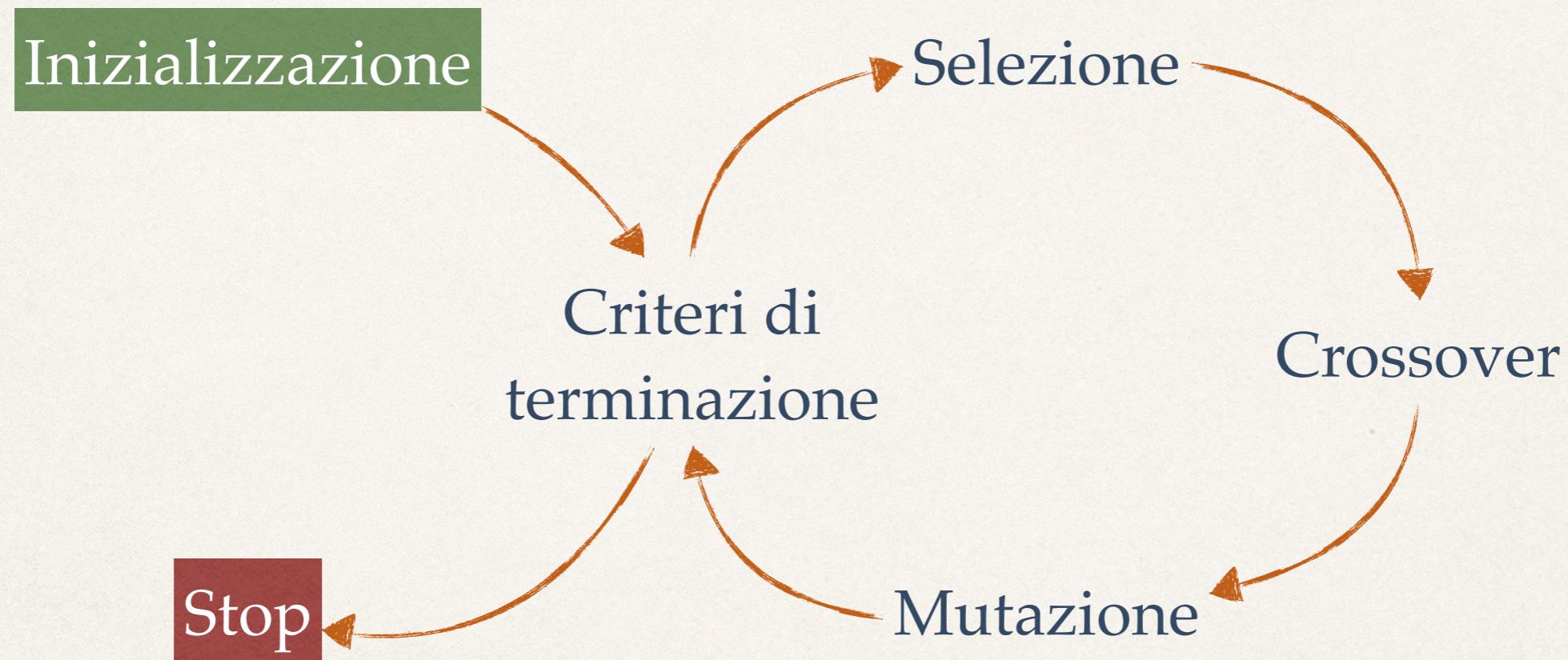
Genotipo

- ❖ Mutazioni casuali

Algoritmo Evolutivo di Base

- ✿ Evolviamo una **popolazione**
(multi-insieme di soluzioni / individui)
- ✿ A ogni **generazione** (ciclo) svolgiamo i seguenti passi:
 - ◆ Selezione
 - ◆ Crossover (riproduzione)
 - ◆ Mutazione

Il Ciclo dell'Evoluzione



Rappresentazione delle soluzioni

Soluzioni come...

Sequenze di bit

Vettori di valori reali

Programmi

Espressioni

Circuiti

Grammatiche

Algoritmi genetici

Programmazione genetica

Cosa vedremo

- ✿ Soluzioni come stringhe di bit: algoritmi genetici (GA)
- ✿ Soluzioni come alberi: programmazione genetica (GP)
- ✿ Sequenze di numeri reali: Strategie Evolutive (ES)
- ✿ Metodi per la computazione parallela e distribuita

Algoritmi Genetici

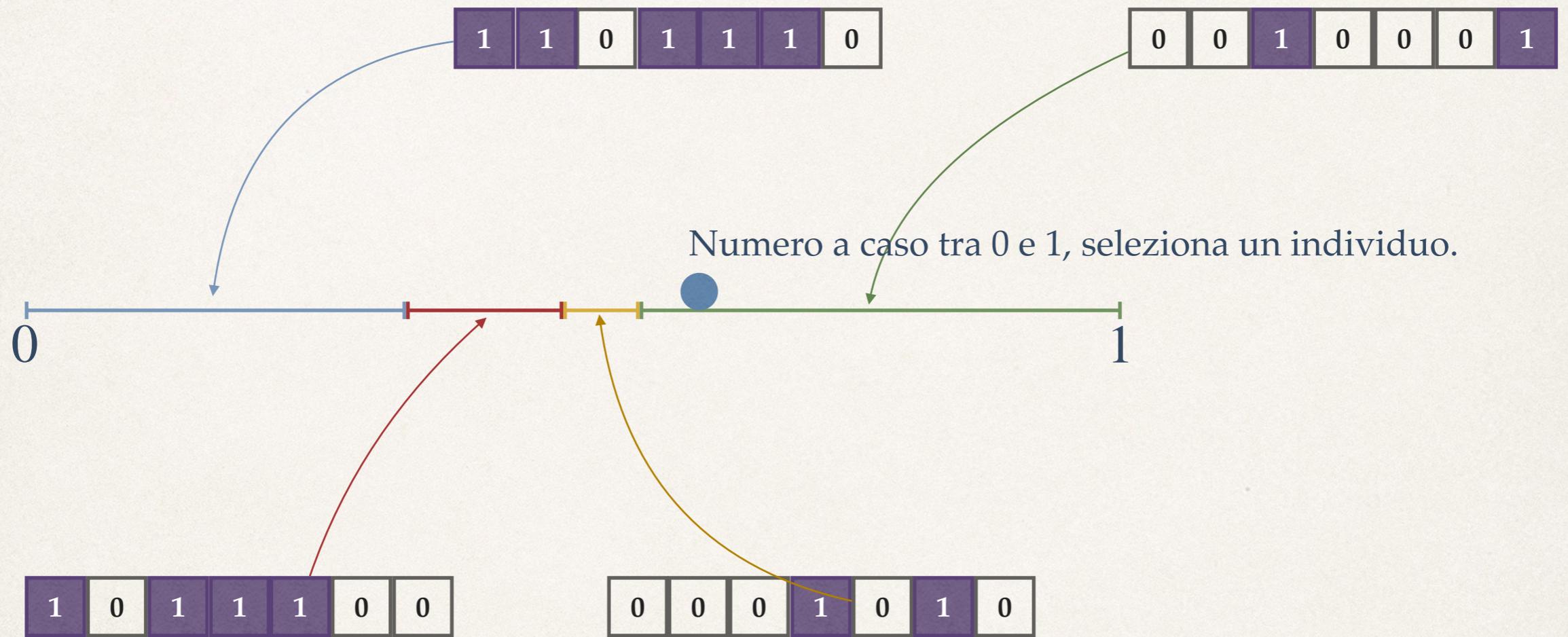
Codifica delle Soluzioni

Ogni soluzione è una sequenza di bit (0 e 1) e può rappresentare:

- ❖ Insiemi (0 = elemento assente, 1 = elemento presente)
- ❖ Numeri interi (scritti in binario)
- ❖ Caratteri e parole
- ❖ Combinazioni dei precedenti
- ❖ etc.

1	0	1	1	1	0	0
---	---	---	---	---	---	---

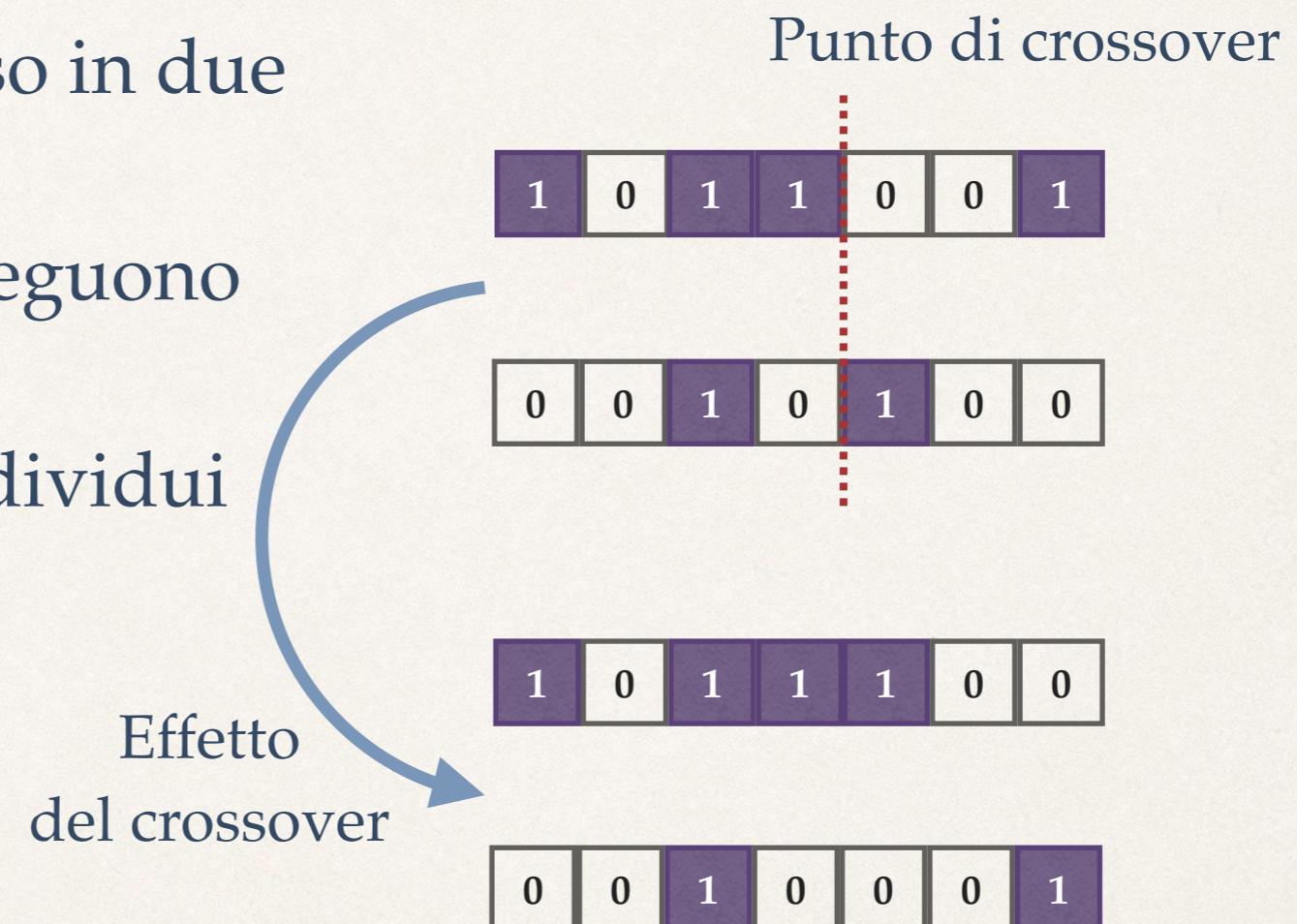
Selezione Roulette Wheel



Ogni individuo ha un segmento di lunghezza proporzionale alla sua fitness rispetto a quella di tutti gli altri individui.

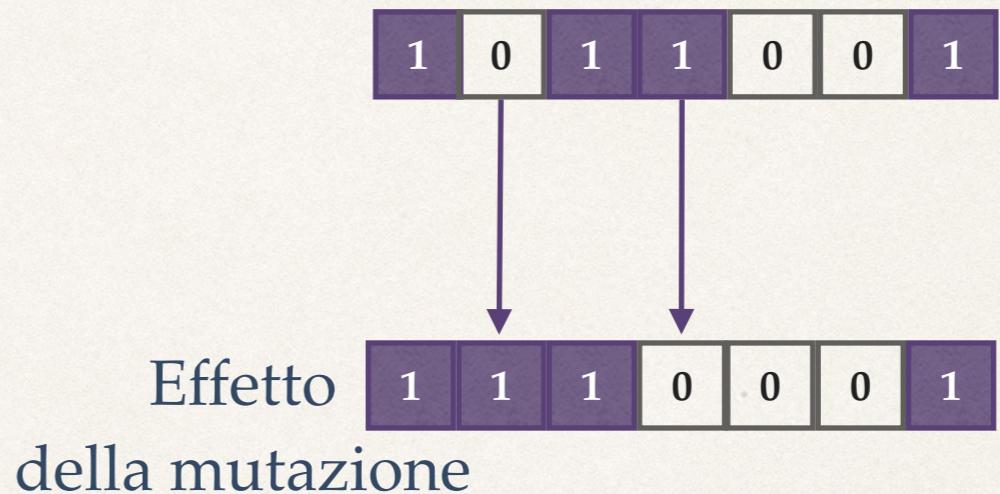
Crossover a un punto

- ❖ Scegliamo un punto a caso in due individui (genitori)
- ❖ Scambiamo le parti che seguono il punto di crossover
- ❖ Generiamo due nuovi individui (figli)
- ❖ Possiamo generalizzare a crossover a n punti



Mutazione

- ❖ La mutazione viene svolta scorrendo tutti i bit dell'individuo
- ❖ Ogni bit viene invertito con una *bassa* probabilità
- ❖ Il risultato è un individuo con alcuni bit cambiati

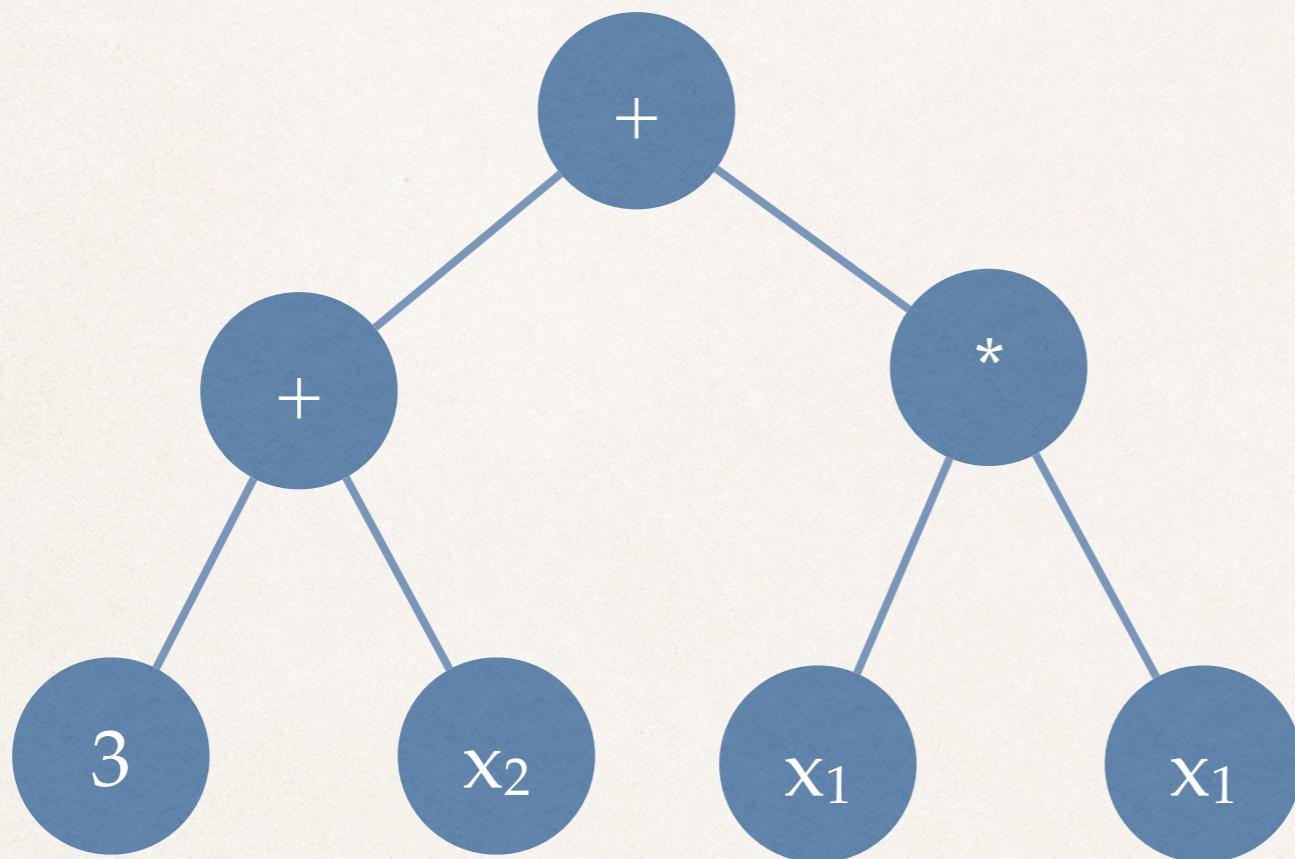


Esempi di Algoritmi Genetici

- ❖ Evoluzione di “mezzi con ruote”:
[https://rednuht.org/genetic cars 2/](https://rednuht.org/genetic_cars_2/)
- ❖ Costruire una creatura virtuale di ossa e muscoli e usare metodi evolutivi per farla correre / saltare:
<https://keiwan.itch.io/evolution>
(esempi di risultati dell’evoluzione:
<http://keiwando.com/evolution/gifs/>)

Programmazione Genetica

Soluzioni come alberi



Genotipo

$$(3 + x_2) + (x_1 \times x_1)$$

Fenotipo

Codifica di una soluzione

$$\mathcal{F} = \{ +, -, \times, \div \}$$

Un insieme di **simboli funzionali** $\mathcal{F} = \{ \wedge, \vee, \neg, \oplus \}$

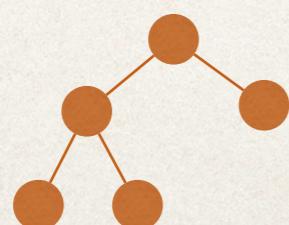
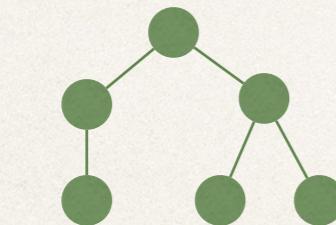
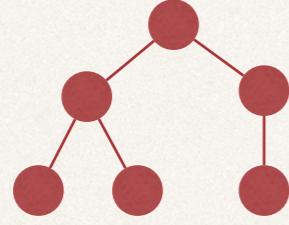
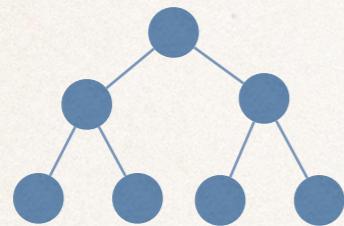
$$\mathcal{F} = \{ \text{if}, \text{set}, \text{sum}, \dots \}$$

{true, false}

Un insieme di **simboli terminali** $\mathcal{T} = \{x_1, \dots, x_n\} \cup \{0, 1, -1, 2, -2, \dots\}$

{0.1, 0.05, 10^{-4} , ...}

Fitness Roulette Wheel



Fitness

2

3

1

4

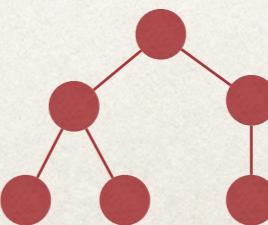
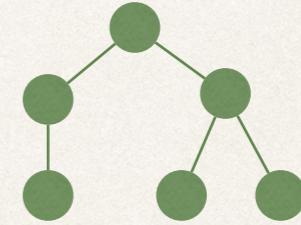
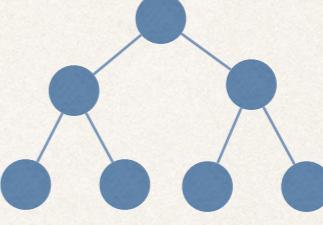
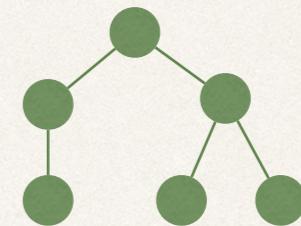
Probabilità

0.3

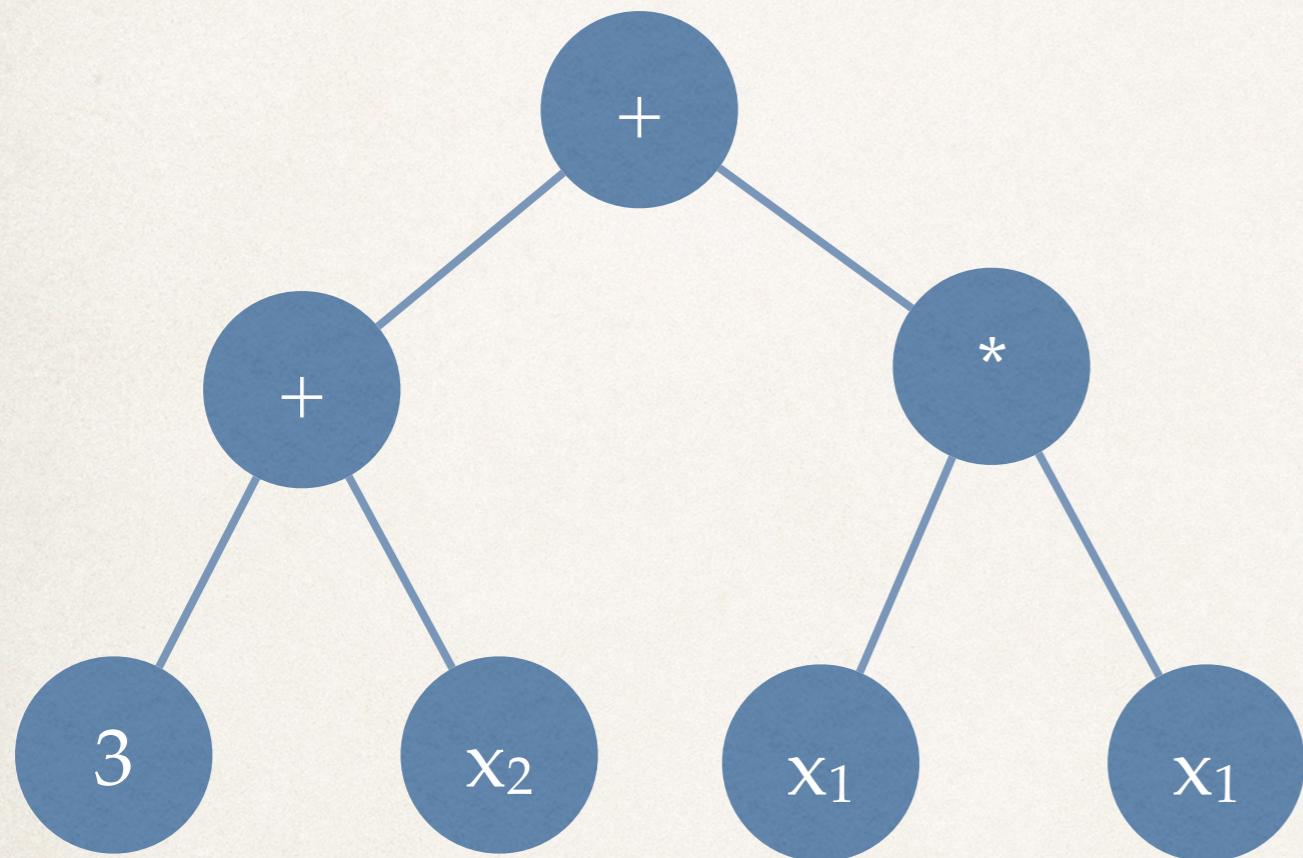
0.2

0.4

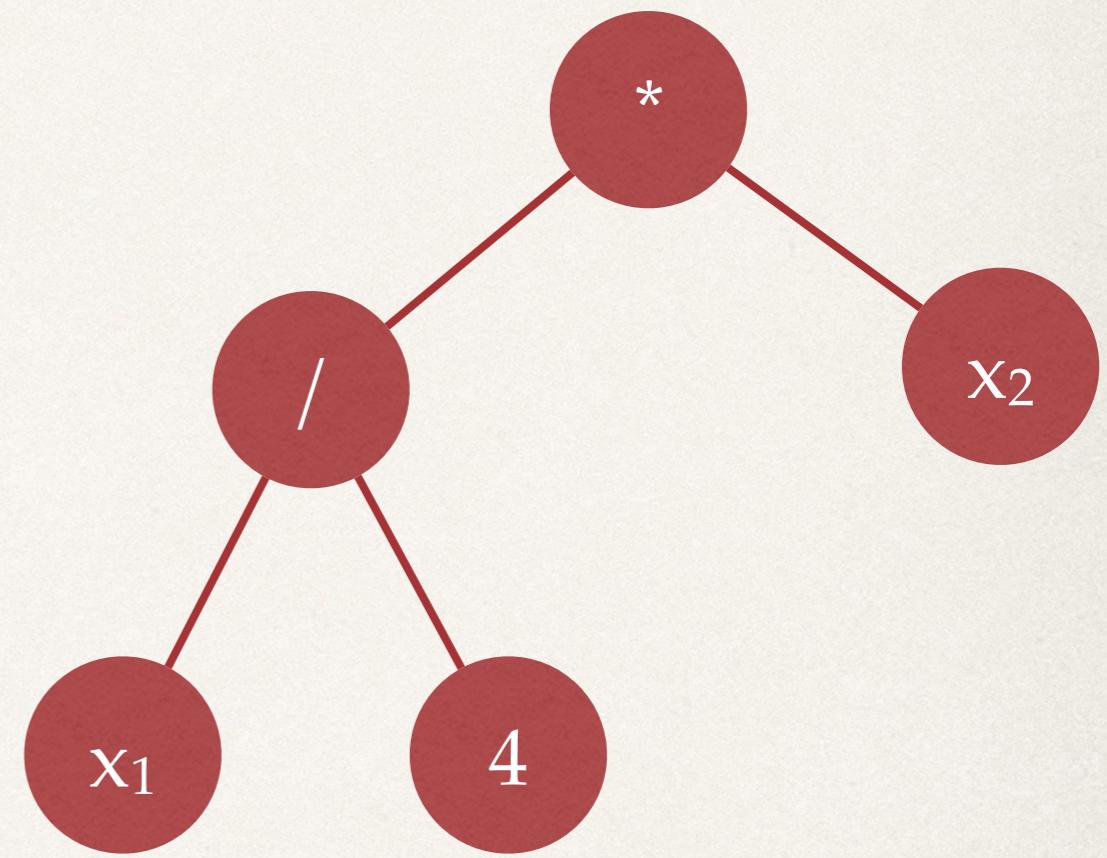
0.1



Crossover di Sottoalberi

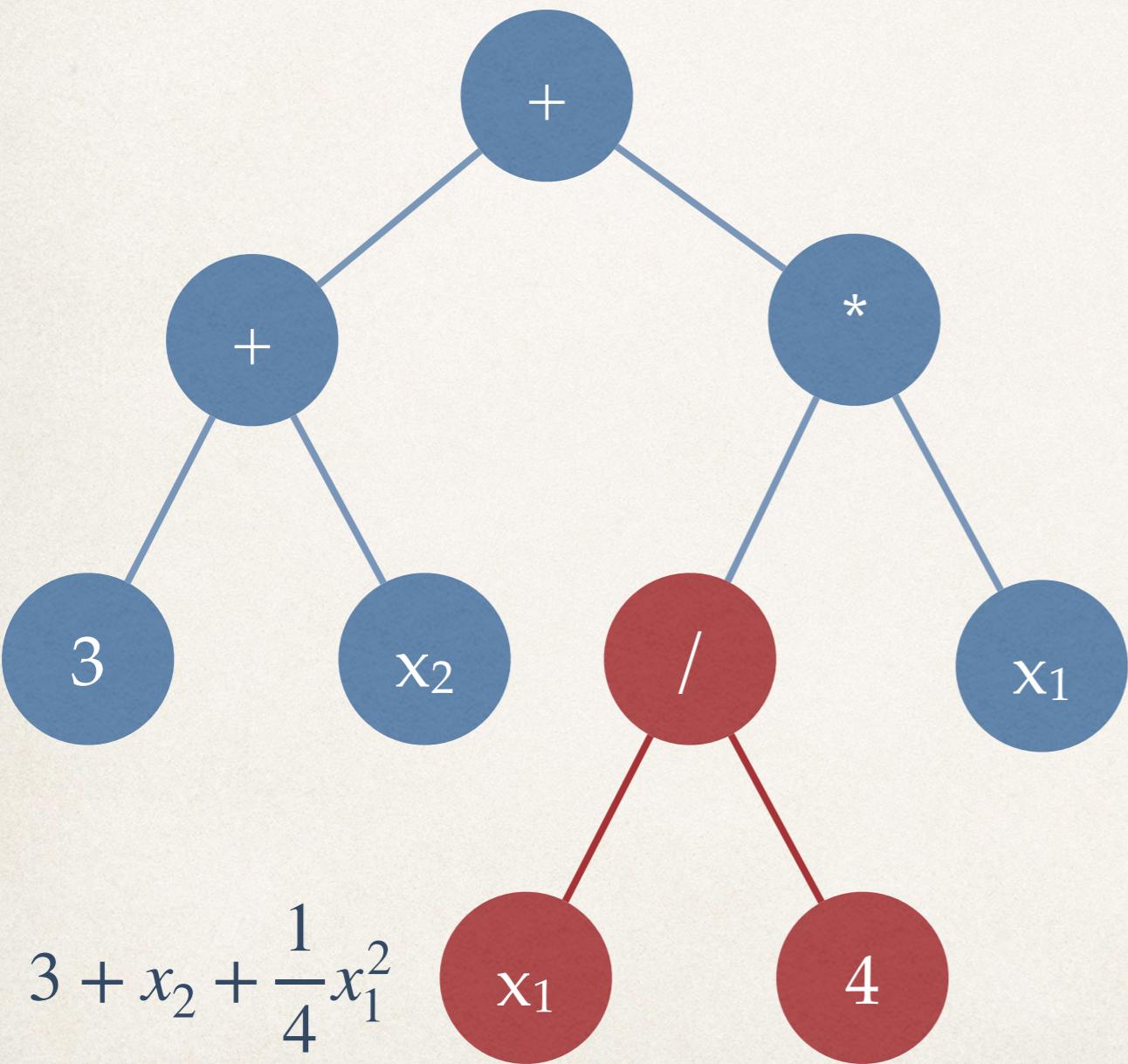


$$3 + x_2 + x_1^2$$

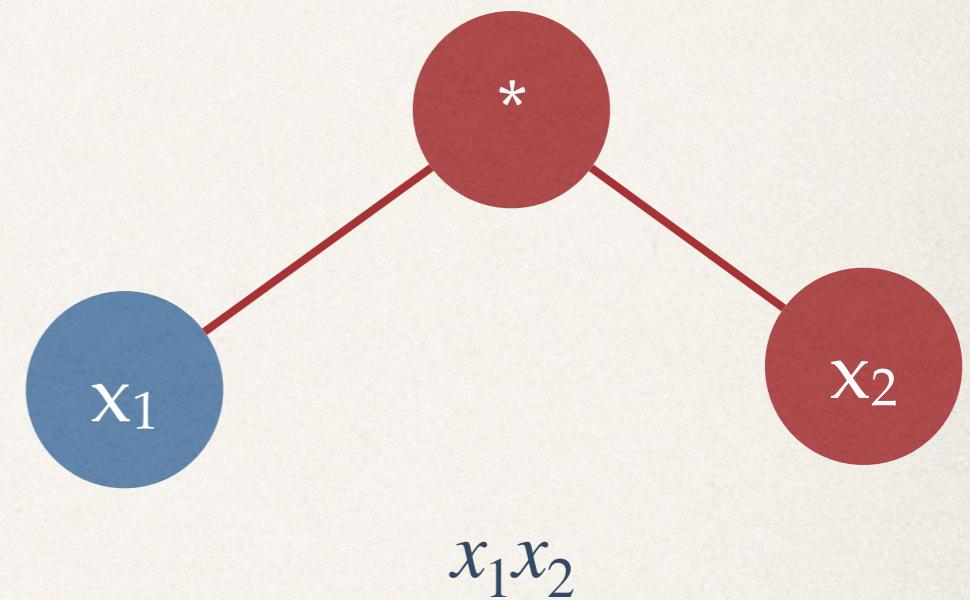


$$\frac{1}{4}x_1x_2$$

Crossover di Sottoalberi

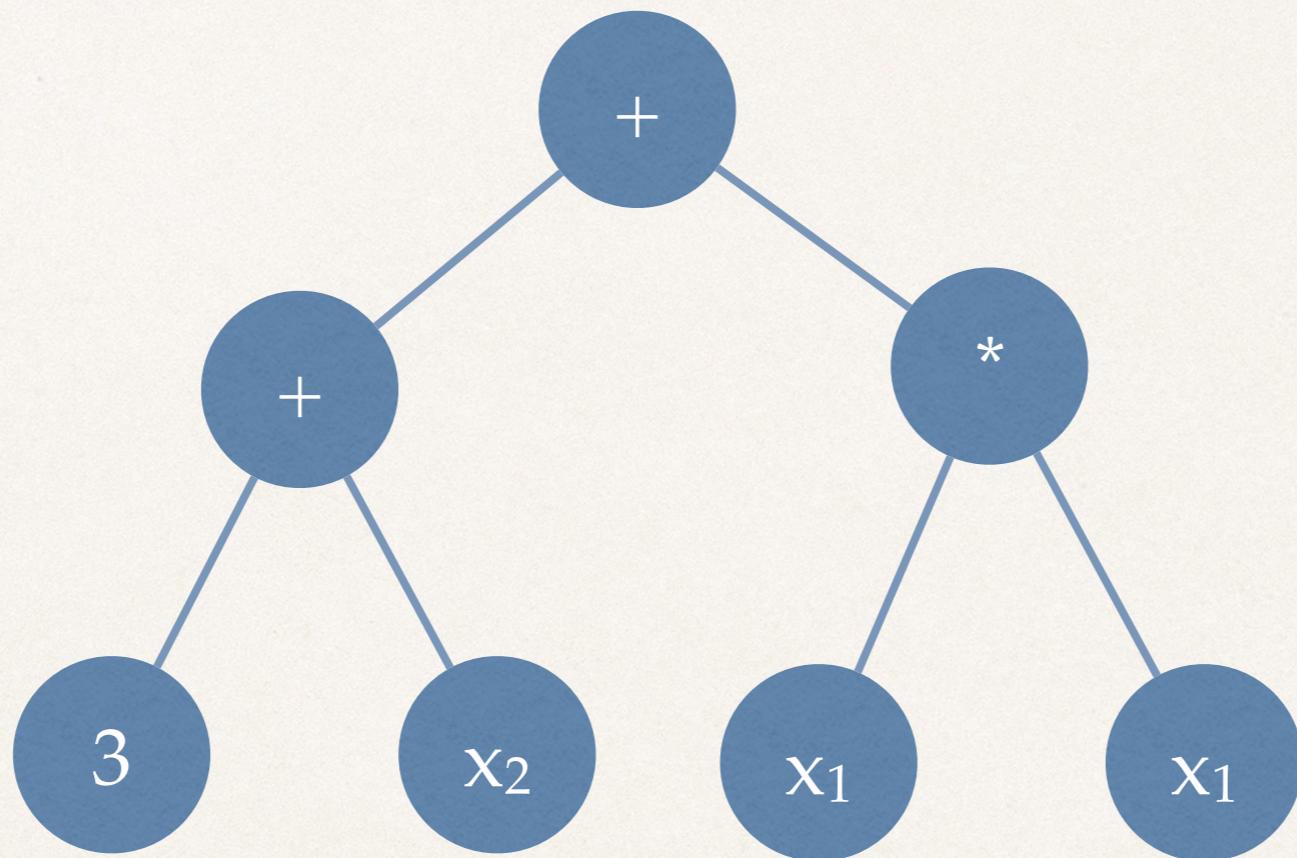


$$3 + x_2 + \frac{1}{4}x_1^2$$



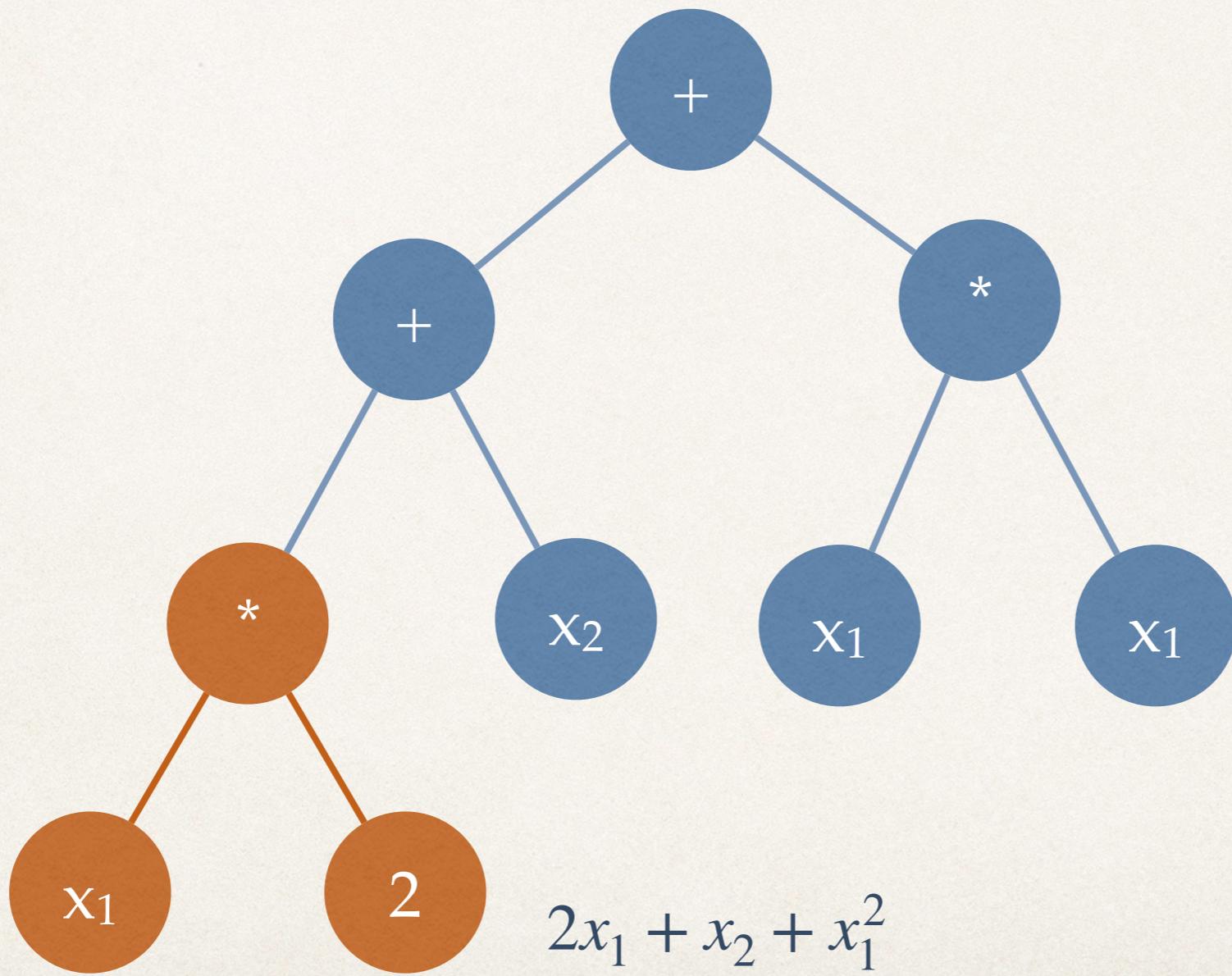
$$x_1 x_2$$

Mutazione di Sottoalberi

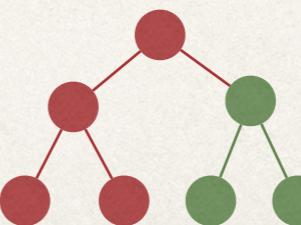
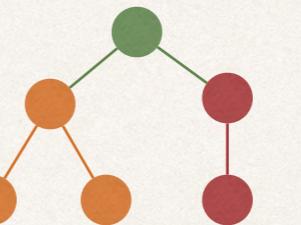
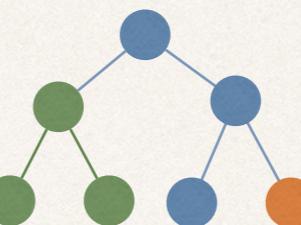
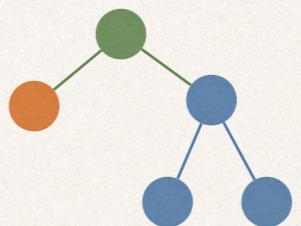
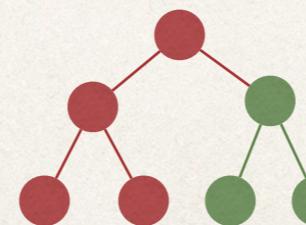
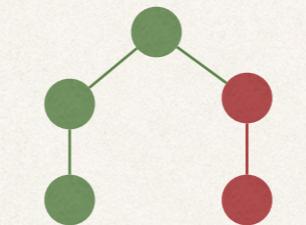
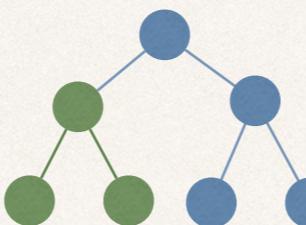
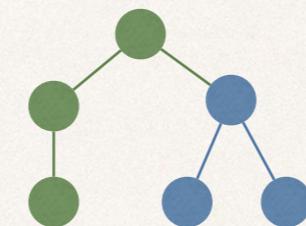
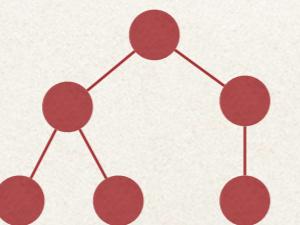
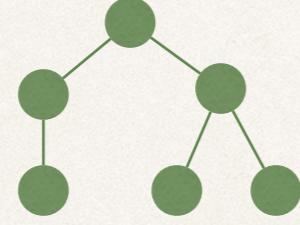
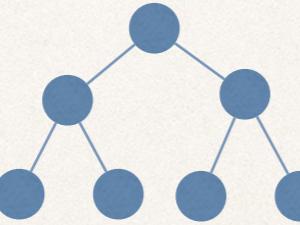
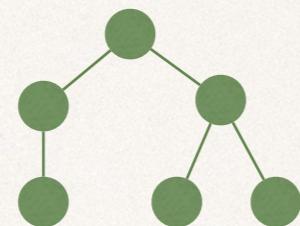
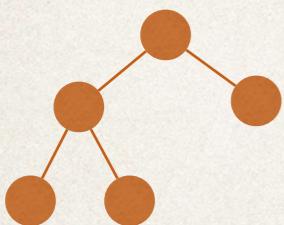
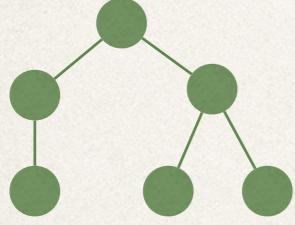
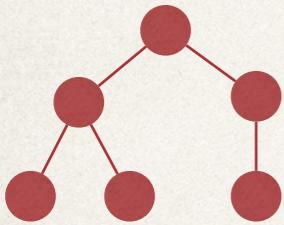
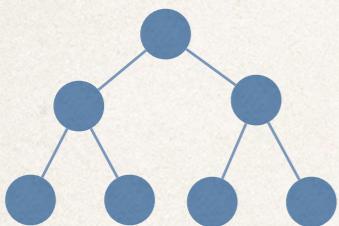


$$3 + x_2 + x_1^2$$

Mutazione di Sottoalberi



Una generazione di GP



Selezione
Crossover
Mutazione

Strategie Evolutive

Strategie Evolutive

- ✿ Ideate negli anni '60
- ✿ Simili agli algoritmi genetici:
 - ✿ Esiste una popolazione di soluzioni
 - ✿ I figli sono derivati dalla mutazione
 - ✿ Esiste un processo di selezione

Strategie Evolutive

- ✿ Ci sono però delle differenze sostanziali:
 - ✿ Non c'è crossover
 - ✿ La selezione è per troncamento (sopravvivono i migliori individui, non è un processo probabilistico)
 - ✿ Solitamente gli individui sono sequenze di numeri floating point (numeri con virgola)

Parametri delle ES



Numero di figli generati

Numero di individui
selezionati



Due tipologie di ES:

$(\mu, \lambda) - ES$

$(\mu + \lambda) - ES$

Ciclo delle (μ, λ) – ES

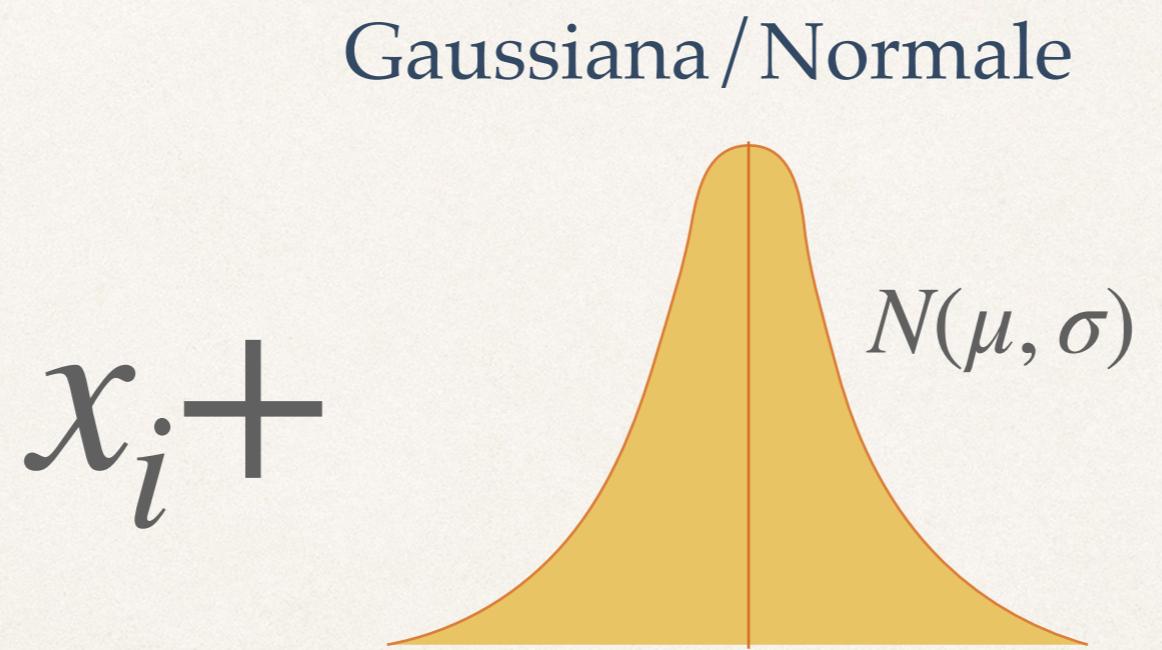


Ciclo delle $(\mu + \lambda)$ – ES



Mutazione

La mutazione è solitamente definita aggiungendo un rumore Gaussiano ad ogni coordinata del vettore



Metodi paralleli e distribuiti

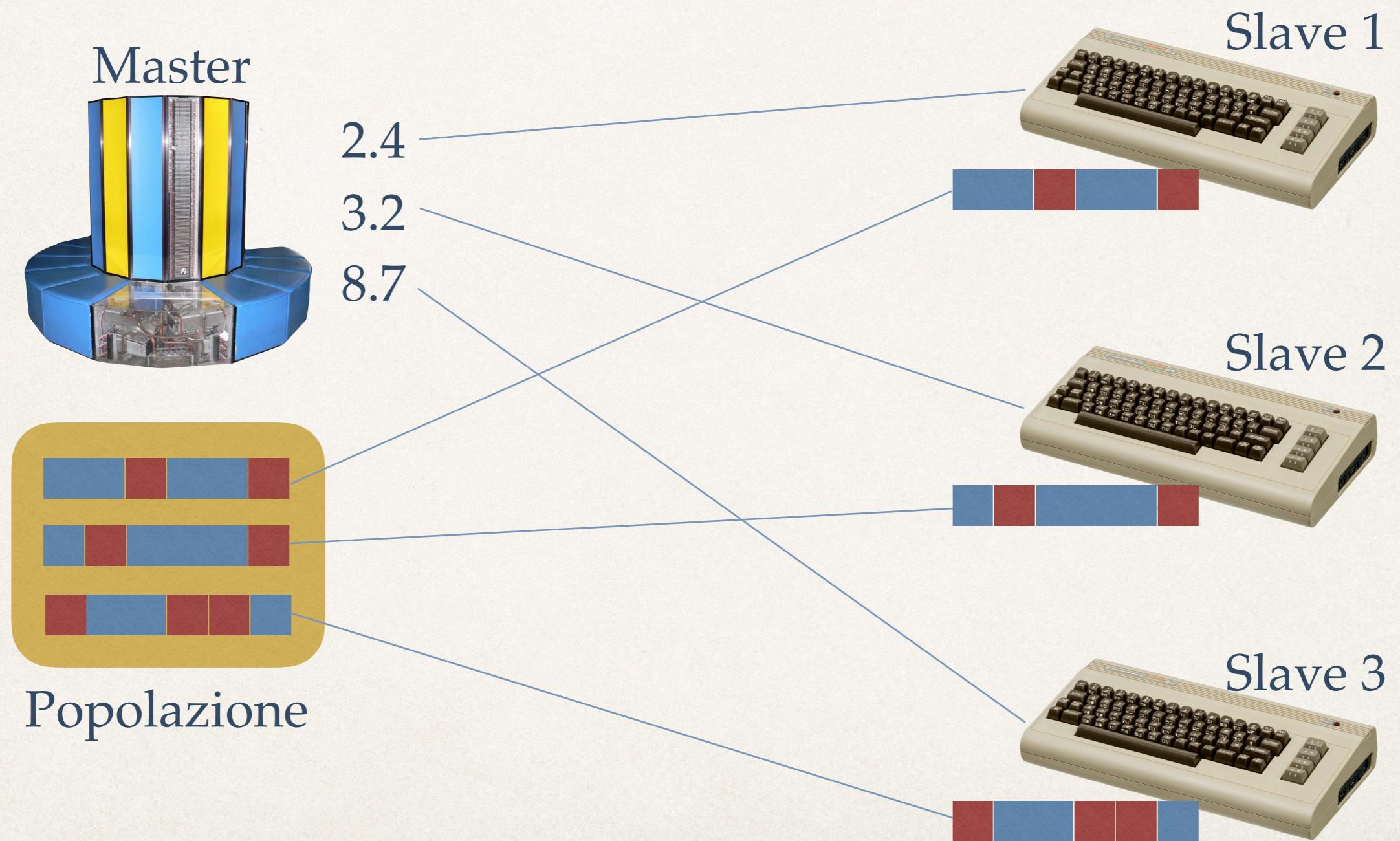
Motivazioni

- ❖ Far girare gli algoritmi evolutivi può essere costoso dal punto di vista delle risorse computazionali
- ❖ Dato che sono basati su una popolazione, spesso la parallelizzazione può essere svolta in modo abbastanza semplice

Valutazione distribuita della fitness

- ✿ Conosciuta anche come approccio master-slave
- ✿ La valutazione della fitness può essere la parte più costosa di un algoritmo evolutivo
- ✿ Si mantiene l'evoluzione in un solo computer / processo, il master
- ✿ La valutazione della fitness è distribuita nei nodi slave

Valutazione distribuita della fitness



Vantaggi e Svantaggi

- ✿ Se la valutazione della fitness è la parte più costosa questo metodo funziona bene
- ✿ È facile aggiungere nodi quando necessario
- ✿ Serve solo trasmettere gli individui ai vari nodi slave...
- ✿ ...ma se il tempo di trasmissione è troppo elevato potremmo non guadagnare in termini di tempo

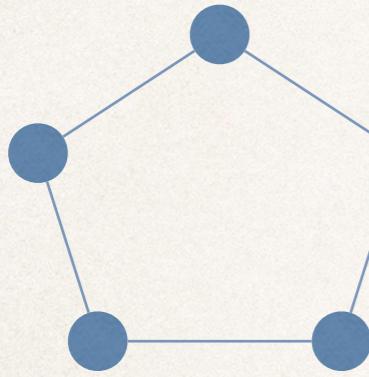
Il Modello a Isole

- ✿ Evoluzione distribuita: come nelle vere isole, ci sono popolazioni che evolvono in modo indipendente e occasionalmente scambiano individui (migrazione)
- ✿ Abbiamo dei parametri aggiuntivi:
 - ✿ Numero delle isole
 - ✿ Topologia delle isole (come sono collegate)

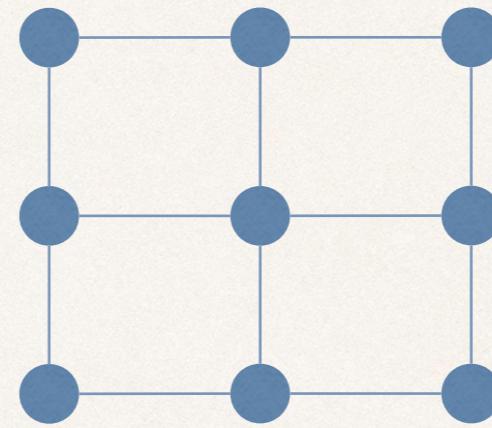
Il Modello a Isole



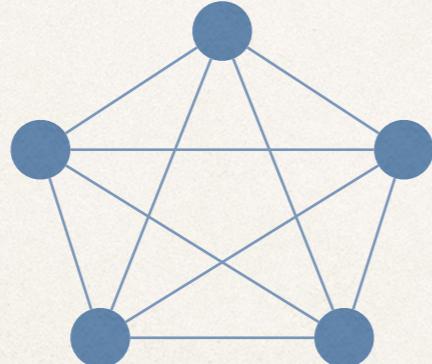
Topologie nel Modello a Isole



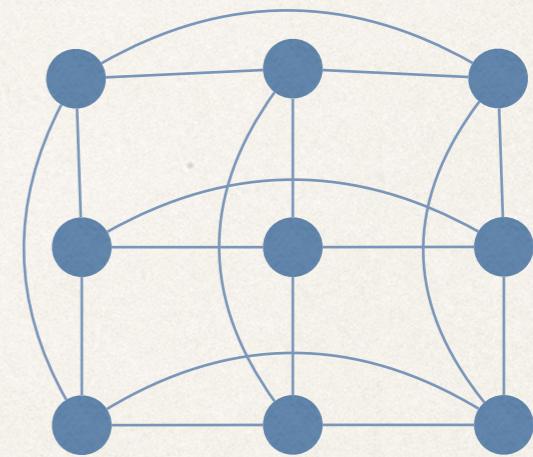
Anello



Griglia



Completamente
connesso



Toroide