Esercitazione 3: Algoritmi genetici

Fondamenti di Intelligenza Artificiale

SAPIENZA
UNIVERSITÀ DI ROMA

A.A. 2022/2023

DIAG, Dipartimento di Ingegneria Informatica, Automatica, e Gestionale "Antonio Ruberti".

Repository codice esercitazioni

link: https://github.com/KRLGroup/FondamentiIA-2223

Download della repo con git (opzionale):

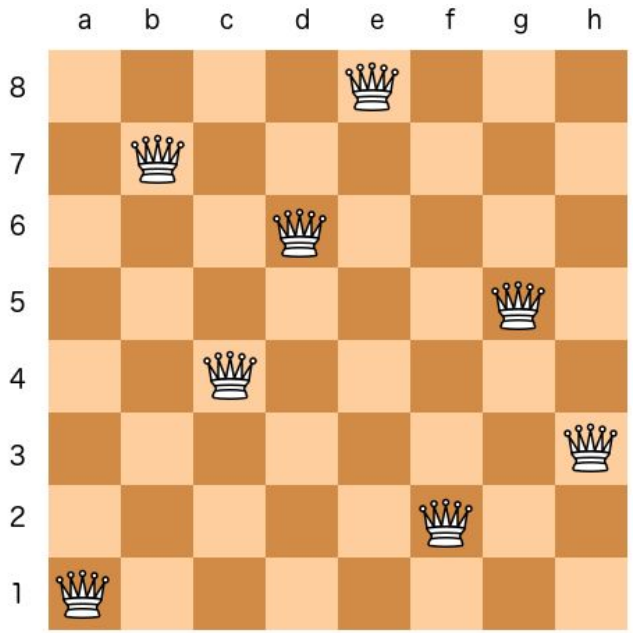
```
git clone https://github.com/KRLGroup/FondamentiIA-2223.git
```

Il codice per questa esercitazione è in “`esercitazione03.py`”

Problema: N-Queens

Data una scacchiera NxN, posizionare N regine sulla scacchiera in maniera che nessuna coppia di regine possa attaccarsi, i.e.:

- occupi la stessa riga
- occupi la stessa colonna
- sia sulla stessa diagonale



Esercizio 1

Completare l'implementazione di un algoritmo genetico per risolvere N-queens. La struttura dell'algoritmo è già presente, e vanno solo completate le fasi di:

1. selezione stocastica dei genitori
2. generazione dei figli tramite crossover
3. mutazione dei figli

Esercizio 1: Rappresentazione degli individui

Ogni individuo è rappresentato come una lista di lunghezza N, contenente interi (*geni*) nell'intervallo [0, …, N-1]. Dato un individuo x, ogni gene x[i] rappresenta una regina posizionata alla riga x[i] e alla colonna i.

Nota: gli indici di righe e colonne partono da 0.

Esercizio 1: funzione di fitness

Implementare la funzione di fitness f(x) che ritorni, per l'individuo x, il numero di coppie di regine **distinte** che non possono attaccarsi.

Nota: le coppie (p, q) e (q, p) non sono distinte, e vanno quindi contate una volta sola.

Esercizio 1: selezione dei parent

Implementare selezione dei genitori con strategia *roulette wheel*: ogni genitore viene estratto dalla popolazione con probabilità proporzionale alla sua fitness.

Esercizio 1: crossover

D) Implementare l'operatore di crossover: dati due genitori x ed y:

1. estrarre un *punto di crossover c* tra 0 e N-1
2. generare il primo figlio come:

   [x[0], …, x[c], y[c+1], …, y[N]]
3. generare il secondo figlio come:

   [y[0], …, y[c], x[c+1], …, x[N]]

Esercizio 1: mutazione

C) Implementare l'operatore di mutazione: dato un individuo x e il *mutation rate r,* ritornare:

- con probabilità *r*, la versione mutata di x
- con probabilità (1-*r*), una copia di x

Per mutare x, estrarre l'indice di un singolo gene da modificare ed estrarre un nuovo valore per quel gene.

Esercizio 2

A) Usando la funzione `run_ga_n_queens` che applica l'algoritmo genetico a una popolazione iniziale fissata, implementare una funzione che stima il success rate dell'algoritmo genetico su **almeno** 25 esperimenti (il numero massimo di generazioni è già fissato).

Il codice già presente stamperà in output il risultato per N in [4,5,6,7,8]

Esercizio 2

B) Come varia il success rate all’aumentare di N? Ipotizzare una possibile spiegazione di questo risultato.

C) Fornire una possibile soluzione al problema riscontrato in B) usando una metodologia utilizzata in un altro algoritmo di ricerca locale.