nQueens-GeneticAlgorithm

November 17, 2023

[ ]:
```
import random
import numpy
from deap import base
from deap import creator
from deap import algorithms
from deap import tools

creator.create('FitnessMax', base.Fitness, weights=(1.0,))
creator.create('Queens', list, fitness=creator.FitnessMax)

def fitness_function(solucao):
        melhorFitness = (len(solucao) * (len(solucao) - 1) // 2)
        h = 0
        #Contagem de ataques na diagonal e vertical
        for i in range(0, len(solucao)):
            for j in range(0, len(solucao)):
                if j > i:
                    # Avalia a diferença entre as colunas e as posições
                    # das rainhas dentro da coluna
                    if abs(i - j) == abs(solucao[i] - solucao[j]):
                       #  print(f'{i} - {j} - {solucao[i]}-{solucao[j]}')
                       h += 1
                    # Ataques por linha (horizontal)
                    # Avalia apenas as posições das rainhas
                    if abs(solucao[i] - solucao[j]) == 0:
                       h += 1
        return (melhorFitness - h),

def n_queens_comb1(n_queens):

    toolbox = base.Toolbox()
    toolbox.register('number_queens', random.randint, 0, n_queens)
    toolbox.register('solution', tools.initRepeat, creator.Queens, toolbox.
 ↪number_queens, n=(n_queens + 1))
    toolbox.register('population',tools.initRepeat, list, toolbox.solution)

    toolbox.register('evaluate', fitness_function)
```

```
    toolbox.register('mate', tools.cxTwoPoint)
    toolbox.register('mutate', tools.mutShuffleIndexes, indpb=0.05)
    toolbox.register('select', tools.selRoulette)
    population = toolbox.population(n=1000)

    CXPB, MUTPB, NGEN = 0.8, 0.01, 100
    hof = tools.HallOfFame(10)
    stats = tools.Statistics(key=lambda ind: ind.fitness.values)
    stats.register('Melhor Solução', numpy.max)
    stats.register('Media', numpy.mean)

    finalPop, logbook = algorithms.eaSimple(population, toolbox, CXPB, MUTPB,␣
↪NGEN, stats=stats, halloffame=hof, verbose=False)

    best_ind = tools.selBest(finalPop, 10)
    for i in best_ind:
        print(f'Solucao: {i} - Fitness {i.fitness}')
```

```
/home/hub/Documents/Disciplina IA/Projeto 1 - Genetic
Algorithm/GenAlgVenv/lib/python3.8/site-packages/deap/creator.py:185:
RuntimeWarning: A class named 'FitnessMax' has already been created and it will
be overwritten. Consider deleting previous creation of that class or rename it.
  warnings.warn("A class named '{0}' has already been created and it "
/home/hub/Documents/Disciplina IA/Projeto 1 - Genetic
Algorithm/GenAlgVenv/lib/python3.8/site-packages/deap/creator.py:185:
RuntimeWarning: A class named 'Queens' has already been created and it will be
overwritten. Consider deleting previous creation of that class or rename it.
  warnings.warn("A class named '{0}' has already been created and it "
```

[ ]:
```
queens = [10,15,20]

for item in queens:
    print('Para', item, 'rainhas:')
    n_queens_comb1(item)
```

```
Para 10 rainhas:
Solucao: [2, 10, 7, 10, 1, 9, 4, 0, 0, 3, 6] - Fitness (53.0,)
Solucao: [5, 2, 10, 6, 10, 9, 4, 0, 3, 9, 7] - Fitness (52.0,)
Solucao: [5, 10, 2, 6, 1, 9, 7, 0, 9, 3, 8] - Fitness (52.0,)
Solucao: [2, 10, 7, 7, 10, 3, 1, 6, 9, 9, 2] - Fitness (51.0,)
Solucao: [5, 1, 2, 6, 10, 6, 2, 0, 7, 3, 0] - Fitness (51.0,)
Solucao: [5, 10, 4, 6, 10, 3, 1, 6, 0, 3, 7] - Fitness (51.0,)
Solucao: [10, 10, 0, 5, 1, 9, 1, 8, 4, 3, 7] - Fitness (51.0,)
Solucao: [5, 1, 2, 8, 10, 3, 1, 6, 9, 0, 6] - Fitness (51.0,)
Solucao: [5, 8, 2, 6, 1, 9, 7, 0, 3, 3, 9] - Fitness (51.0,)
Solucao: [5, 8, 2, 6, 10, 10, 4, 0, 9, 3, 7] - Fitness (51.0,)
Para 15 rainhas:
```

```
Solucao: [1, 3, 8, 0, 4, 15, 15, 9, 11, 0, 2, 7, 6, 14, 12, 2] - Fitness
(114.0,)
Solucao: [13, 0, 7, 9, 2, 15, 3, 4, 10, 12, 5, 3, 14, 13, 8, 1] - Fitness
(114.0,)
Solucao: [13, 6, 8, 8, 11, 14, 3, 0, 0, 10, 4, 1, 13, 15, 12, 2] - Fitness
(113.0,)
Solucao: [12, 10, 6, 11, 2, 1, 11, 0, 5, 15, 4, 8, 13, 9, 12, 7] - Fitness
(112.0,)
Solucao: [3, 5, 8, 1, 15, 2, 14, 12, 10, 0, 11, 12, 11, 3, 0, 2] - Fitness
(112.0,)
Solucao: [2, 6, 14, 11, 6, 15, 13, 8, 3, 0, 8, 3, 9, 7, 12, 15] - Fitness
(112.0,)
Solucao: [12, 10, 6, 11, 2, 1, 11, 0, 5, 15, 4, 8, 13, 9, 12, 7] - Fitness
(112.0,)
Solucao: [13, 6, 8, 11, 15, 10, 3, 4, 0, 0, 4, 7, 8, 12, 2, 1] - Fitness
(111.0,)
Solucao: [12, 10, 10, 1, 1, 15, 11, 6, 11, 13, 0, 7, 14, 13, 15, 7] - Fitness
(111.0,)
Solucao: [15, 10, 5, 13, 4, 12, 11, 0, 2, 14, 2, 7, 8, 15, 8, 3] - Fitness
(111.0,)
Para 20 rainhas:
Solucao: [5, 16, 2, 0, 3, 9, 17, 18, 16, 7, 12, 20, 14, 8, 6, 9, 1, 10, 1, 7, 4]
- Fitness (199.0,)
Solucao: [11, 1, 20, 0, 14, 4, 15, 17, 20, 16, 0, 9, 3, 12, 17, 2, 18, 3, 14, 8,
8] - Fitness (199.0,)
Solucao: [20, 5, 15, 16, 3, 11, 16, 1, 20, 1, 14, 10, 0, 9, 11, 13, 2, 18, 7, 8,
6] - Fitness (198.0,)
Solucao: [10, 4, 15, 12, 2, 0, 7, 18, 16, 3, 6, 20, 19, 14, 20, 3, 13, 3, 18,
13, 5] - Fitness (198.0,)
Solucao: [10, 7, 11, 16, 8, 1, 20, 4, 20, 0, 0, 13, 7, 12, 16, 9, 13, 3, 19, 12,
6] - Fitness (197.0,)
Solucao: [11, 5, 2, 4, 2, 19, 16, 18, 15, 3, 0, 7, 5, 18, 12, 0, 13, 3, 18, 8,
9] - Fitness (197.0,)
Solucao: [5, 11, 6, 12, 2, 0, 15, 1, 20, 16, 12, 7, 19, 6, 20, 8, 13, 14, 4, 8,
4] - Fitness (197.0,)
Solucao: [15, 20, 3, 16, 12, 17, 0, 4, 11, 0, 0, 2, 14, 12, 20, 7, 10, 3, 5, 6,
9] - Fitness (197.0,)
Solucao: [15, 3, 10, 12, 3, 4, 9, 18, 5, 2, 6, 5, 17, 20, 9, 13, 8, 3, 1, 8, 6]
- Fitness (197.0,)
Solucao: [5, 18, 2, 0, 19, 10, 16, 5, 15, 1, 6, 5, 20, 12, 3, 14, 10, 3, 11, 7,
17] - Fitness (197.0,)
```

[ ]:
```python
def n_queens_comb2(n_queens):

    toolbox = base.Toolbox()
    toolbox.register('number_queens', random.randint, 0, n_queens)
```

```
    toolbox.register('solution', tools.initRepeat, creator.Queens, toolbox.
↪number_queens, n=(n_queens + 1))
    toolbox.register('population',tools.initRepeat, list, toolbox.solution)

    toolbox.register('evaluate', fitness_function)
    toolbox.register('mate', tools.cxUniform, indpb=0.05)
    toolbox.register('mutate', tools.mutUniformInt, indpb=0.05, low=1, up=5)
    toolbox.register('select', tools.selTournament, tournsize = 3)
    population = toolbox.population(n=1000)

    CXPB, MUTPB, NGEN = 0.8, 0.01, 100
    hof = tools.HallOfFame(10)
    stats = tools.Statistics(key=lambda ind: ind.fitness.values)
    stats.register('Melhor Solução', numpy.max)
    stats.register('Media', numpy.mean)

    finalPop, logbook = algorithms.eaSimple(population, toolbox, CXPB, MUTPB,␣
↪NGEN, stats=stats, halloffame=hof, verbose=False)

    best_ind = tools.selBest(finalPop, 10)
    for i in best_ind:
        print(f'Solucao: {i} - Fitness {i.fitness}')
```

[ ]:
```
queens = [10,15,20]

for item in queens:
    print('Para', item, 'rainhas:')
    n_queens_comb2(item)
```

```
Para 10 rainhas:
Solucao: [4, 2, 7, 3, 10, 6, 9, 0, 5, 1, 8] - Fitness (54.0,)
Solucao: [4, 2, 7, 3, 10, 6, 9, 0, 5, 1, 8] - Fitness (54.0,)
Solucao: [4, 2, 7, 3, 10, 6, 9, 0, 5, 1, 8] - Fitness (54.0,)
Solucao: [4, 2, 7, 3, 10, 6, 9, 0, 5, 1, 8] - Fitness (54.0,)
Solucao: [4, 2, 7, 3, 10, 6, 9, 0, 5, 1, 8] - Fitness (54.0,)
Solucao: [4, 2, 7, 3, 10, 6, 9, 0, 5, 1, 8] - Fitness (54.0,)
Solucao: [4, 2, 7, 3, 10, 6, 9, 0, 5, 1, 8] - Fitness (54.0,)
Solucao: [4, 2, 7, 3, 10, 6, 9, 0, 5, 1, 8] - Fitness (54.0,)
Solucao: [4, 2, 7, 3, 10, 6, 9, 0, 5, 1, 8] - Fitness (54.0,)
Solucao: [4, 2, 7, 3, 10, 6, 9, 0, 5, 1, 8] - Fitness (54.0,)
Para 15 rainhas:
Solucao: [7, 13, 1, 14, 6, 3, 0, 4, 11, 9, 12, 2, 8, 2, 15, 10] - Fitness
(118.0,)
Solucao: [7, 13, 1, 14, 6, 3, 0, 4, 11, 9, 12, 2, 8, 2, 15, 10] - Fitness
(118.0,)
Solucao: [7, 13, 1, 14, 6, 3, 0, 4, 11, 9, 12, 2, 8, 2, 15, 10] - Fitness
(118.0,)
```

Solucao: [7, 13, 1, 14, 6, 3, 0, 4, 11, 9, 12, 2, 8, 2, 15, 10] - Fitness (118.0,)
Solucao: [7, 13, 1, 14, 6, 3, 0, 4, 11, 9, 12, 2, 8, 2, 15, 10] - Fitness (118.0,)
Solucao: [7, 13, 1, 14, 6, 3, 0, 4, 11, 9, 12, 2, 8, 2, 15, 10] - Fitness (118.0,)
Solucao: [7, 13, 1, 14, 6, 3, 0, 4, 11, 9, 12, 2, 8, 2, 15, 10] - Fitness (118.0,)
Solucao: [7, 13, 1, 14, 6, 3, 0, 4, 11, 9, 12, 2, 8, 2, 15, 10] - Fitness (118.0,)
Solucao: [7, 13, 1, 14, 6, 3, 0, 4, 11, 9, 12, 2, 8, 2, 15, 10] - Fitness (118.0,)
Solucao: [7, 13, 1, 14, 6, 3, 0, 4, 11, 9, 12, 2, 8, 2, 15, 10] - Fitness (118.0,)
Para 20 rainhas:
Solucao: [2, 14, 2, 4, 7, 9, 20, 18, 1, 8, 17, 5, 0, 11, 19, 6, 3, 12, 10, 13, 16] - Fitness (208.0,)
Solucao: [2, 14, 2, 4, 7, 9, 20, 18, 1, 8, 17, 5, 0, 11, 19, 6, 3, 12, 10, 13, 16] - Fitness (208.0,)
Solucao: [2, 14, 2, 4, 7, 9, 20, 18, 1, 8, 17, 5, 0, 11, 19, 6, 3, 12, 10, 13, 16] - Fitness (208.0,)
Solucao: [2, 14, 2, 4, 7, 9, 20, 18, 1, 8, 17, 5, 0, 11, 19, 6, 3, 12, 10, 13, 16] - Fitness (208.0,)
Solucao: [2, 14, 2, 4, 7, 9, 20, 18, 1, 8, 17, 5, 0, 11, 19, 6, 3, 12, 10, 13, 16] - Fitness (208.0,)
Solucao: [2, 14, 2, 15, 7, 9, 20, 18, 1, 8, 17, 5, 0, 11, 19, 6, 3, 12, 10, 13, 16] - Fitness (208.0,)
Solucao: [2, 14, 2, 4, 7, 9, 20, 18, 1, 8, 17, 5, 0, 11, 19, 6, 3, 12, 10, 13, 16] - Fitness (208.0,)
Solucao: [2, 14, 2, 4, 7, 9, 20, 18, 1, 8, 17, 5, 0, 11, 19, 6, 3, 12, 10, 13, 16] - Fitness (208.0,)
Solucao: [2, 14, 2, 4, 7, 9, 20, 18, 1, 8, 17, 5, 0, 11, 19, 6, 3, 12, 10, 13, 16] - Fitness (208.0,)
Solucao: [2, 14, 2, 4, 7, 9, 20, 18, 1, 8, 17, 5, 0, 11, 19, 6, 3, 12, 10, 13, 16] - Fitness (208.0,)