Documentação Trabalho de Implementação - Meta-heurística

Vítor Gabriel Reis Caitité

1 - Algoritmo Genético (GA)

O Algoritmo Genético (GA) foi a meta-heurística selecionada para implementação neste trabalho. Os GAs são algoritmos matemáticos inspirados nos mecanismos de evolução natural e recombinação genética. Esse algoritmo fornece um mecanismo de busca adaptativa que se baseia no princípio Darwiniano de reprodução e sobrevivência dos mais aptos. O pseudocódigo do algoritmo implementado pode ser vasto abaixo.

Algorithm 1 Algoritmo Genético

```
1: Hiper-parâmetros: tamanho da população, taxa de elitismo, taxa de mutação, quantidade de gerações executar, método de
   seleção, eventuais parâmetros do método de seleção.
2: Entrada: Instância contendo coordenadas euclidianas dos nós.
3: Saída: Solução de menor custo da última geração e o seu custo.
4: Inicializa população de indivíduos.
5: while geração $\leq$ max_gerações do
6:     Avalie a população.
7:     Selecione os pais.
8:     Crossover entre os pais selecionados.
9:     Etapa de mutação nos filhos gerados no passo anterior.
10: end while
11: return Solução de custo mínimo da população e seu custo.
```

Para os testes realizados utilizou-se dois modelos de GA:

- GA 1 - Algoritmo utilizando torneio como método de seleção. Esse método consiste em escolher aleatoriamente k indivíduos para um torneio. O vencedor de cada torneio (aquele com melhor *fitness*) é selecionado para realizar o *crossover*. Nessa implementação considerou-se $k = 3$.
- GA 2 - Algoritmo utilizando *roulette wheel* como método de seleção. Nesse tipo de seleção a probabilidade de escolha de um indivíduo para reprodução da próxima geração é proporcional ao seu fitness, quanto melhor o fitness, maior a chance desse indivíduo ser escolhido.

Para ambos os modelos considerou-se um elitismo de 15% e uma taxa de mutação adaptativa. Se o indivíduo tem um *fitness* melhor que a média da população a chance dele sofrer mutação é menor (0.2), caso contrário é maior (0.4). Tanto para o GA 1 quanto para o GA 2 o tamanho da população foi 50 e a população inicial de indivíduos foi gerada através do algoritmo do vizinho mais próximo. Por fim, os algoritmos foram executados por 3000 gerações.

2 - Resultados:

Os algoritmos foram implementado em Python (3.9). E os resultados estão expressos na forma "*média +/- desvio padrão*". O número de testes foi fixado em 10. Os testes foram executados em um notebook com processador i7 7th Gen., 8 Gb de RAM e sistema Linux.

Table 1: Custo obtido por cada algoritmo em cada teste.

Arquivo	GA 1	GA 2
kroA150.tsp	**29087 +/- 405**	30029 +/- 427
kroB100.tsp	**23340 +/- 513**	23823 +/- 386
pr107.tsp	**45174 +/- 47**	45184 +/- 170
kroC100.tsp	**21777 +/- 230**	22614 +/- 361
rat99.tsp	**1266 +/- 11**	1296 +/- 18
st70.tsp	**710 +/- 6**	711 +/- 11
kroB150.tsp	**28732 +/- 603**	30256 +/- 1187
kroB200.tsp	**32669 +/- 527**	34632 +/- 689
pr136.tsp	**106259 +/- 942**	107674 +/- 2208
pr144.tsp	**60765 +/- 16**	60782 +/- 56
pr124.tsp	**62153 +/- 1195**	62986 +/- 1459
pr76.tsp	**114029 +/- 1695**	118220 +/- 3335
kroD100.tsp	**23372 +/- 183**	23945 +/- 600
kroA200.tsp	**31877 +/- 472**	34430 +/- 623
kroE100.tsp	**23301 +/- 285**	23617 +/- 289
lin105.tsp	**15893 +/- 330**	16250 +/- 377
rat195.tsp	**2479 +/- 24**	2734 +/- 108
berlin52.tsp	**8156 +/- 39**	8228 +/- 71
kroA100.tsp	**22572 +/- 314**	22952 +/- 825
att48.tsp	11028 +/- 135	**10829 +/- 123**
pr152.tsp	**76920 +/- 1028**	78085 +/- 1818

Table 2: Tempo de execução (*ms*).

Arquivo	GA 1	GA 2
kroA150.tsp	134040.0 +/- 8178.2	139908.4 +/- 4517.9
kroB100.tsp	65387.0 +/- 2487.4	69732.9 +/- 3363.7
pr107.tsp	67056.3 +/- 3012.5	71435.5 +/- 1988.2
kroC100.tsp	58097.6 +/- 625.4	63398.3 +/- 652.5
rat99.tsp	58947.7 +/- 3166.4	64392.5 +/- 3982.2
st70.tsp	34739.5 +/- 2612.9	40618.3 +/- 3346.9
kroB150.tsp	134949.6 +/- 6182.9	141340.9 +/- 5887.1
kroB200.tsp	216338.2 +/- 16518.6	223231.0 +/- 17543.5
pr136.tsp	100255.2 +/- 1386.4	106167.4 +/- 1285.0
pr144.tsp	109537.8 +/- 1847.6	115373.4 +/- 2381.8
pr124.tsp	83776.4 +/- 1527.4	89298.0 +/- 1998.9
pr76.tsp	34640.3 +/- 592.8	39162.0 +/- 415.6
kroD100.tsp	55290.0 +/- 136.6	60287.2 +/- 258.0
kroA200.tsp	198952.8 +/- 2936.9	207792.0 +/- 3703.9
kroE100.tsp	57509.3 +/- 1109.8	62880.9 +/- 964.2
lin105.tsp	62769.6 +/- 1863.4	67637.8 +/- 1586.2
rat195.tsp	206390.2 +/- 16779.8	219699.6 +/- 18588.7
berlin52.tsp	22090.4 +/- 1227.5	27132.4 +/- 1900.0
kroA100.tsp	64939.0 +/- 2992.9	73462.4 +/- 3435.4
att48.tsp	19787.0 +/- 1819.5	23673.2 +/- 1055.2
pr152.tsp	136481.6 +/- 11074.5	140757.1 +/- 9252.4