INE-5430 - Inteligência Artificial
Trabalho 3 - Algoritmos genéticos
Grupo: Tiago Luz
Charlotte Lesouef

Relatório: Solução do Problema da Mochila Utilizando Algoritmo Genético

1. Introdução

O Problema da Mochila é um problema clássico de otimização combinatória onde o objetivo é selecionar um subconjunto de itens de uma lista, de forma que a soma dos pesos dos itens escolhidos não exceda a capacidade máxima da mochila e que o valor total seja maximizado. Neste projeto, implementamos uma solução utilizando Algoritmos Genéticos (AGs) com a biblioteca PyGAD.

2. Descrição do Problema

Itens e Parâmetros

Para este exemplo, utilizamos os seguintes itens, representados como pares `(peso, valor)`:

```
items = [(3, 5), (5, 8), (6, 9), (7, 13), (4, 7), (2, 4)]
maximum_capacity = 15
```

A capacidade máxima da mochila foi definida como **15 kg**. O objetivo é maximizar o valor total dos itens selecionados sem exceder este limite.

3. Implementação da Solução

Estrutura dos Genes

Cada solução (indivíduo) é representada como uma lista de genes, onde cada gene indica se o item correspondente está incluído na solução:

- `0`: Item não selecionado
- 1: Item selecionado

Exemplo: `[1, 0, 1, 0, 0, 1]` indica que os itens 1, 3 e 6 estão na mochila.

Parâmetros do Algoritmo Genético

Os parâmetros escolhidos para o algoritmo foram definidos da seguinte forma:

- **num_generations=50:** Definimos um número pensando em permitir ao algoritmo evoluir e encontrar soluções de maior qualidade ao longo das iterações.
- **sol_per_pop=10:** O número de soluções (indivíduos) por população foi fixado em **10**.
- **num_parents_mating=5:** O número de pais escolhidos para cruzamento foi **5**, o que significa que metade da população sobrevive para a próxima geração e possibilita uma diversidade razoável de combinações na geração dos descendentes.
- **crossover_type="uniform":** Utilizamos o cruzamento uniforme, onde cada gene do descendente é escolhido aleatoriamente entre os genes dos pais, o que pareceu fazer sentido dado a estrutura dos nossos genes.
- **mutation_type="random":** O tipo de mutação utilizado foi o "random", em que genes são alterados aleatoriamente para aumentar a diversidade genética e evitar estagnação em mínimos locais.
- **mutation_percent_genes=10:** A taxa de mutação foi definida em **10%**, permitindo uma pequena alteração em alguns genes de cada indivíduo, ajudando a explorar novas áreas do espaço de busca.
- **parent_selection_type="rws":** O método de seleção de pais foi o "Roulette Wheel Selection" (RWS), que seleciona indivíduos com probabilidade proporcional ao valor da função fitness, favorecendo indivíduos com fitness superior.
- **gene_space=[0, 1]:** O espaço dos genes foi limitado a `[0, 1]`, devido a escolha binária de cada item.

4. Função Fitness

A função fitness, responsável por avaliar a qualidade de cada solução, foi definida da seguinte forma:

```
def fitness_func(ga_instance, solution, solution_idx):
      weight = sum([v * items[i][0] for i, v in enumerate(solution)])
      value = sum([v * items[i][1] for i, v in enumerate(solution)])
      if weight > maximum_capacity:
            return 0
      fitness = 1 / abs(max_value - value)
            return fitness
```

- **Cálculo do Peso e Valor**: O peso total é calculado somando os produtos dos genes (0 ou 1) pelos pesos dos itens correspondentes. O valor total é calculado somando os produtos dos genes pelos valores dos itens correspondentes.
- **Penalização para Soluções Inválidas**: Se o peso exceder a capacidade máxima da mochila (`maximum_capacity`), o fitness da solução é definido como 0, descartando soluções que não atendem à restrição.
- **Cálculo da Fitness**: O fitness é calculado como `1 / abs(max_value - value)`, onde `max_value` é a soma dos valores de todos os itens. Este cálculo favorece soluções com maior valor total, e a minimização de `abs(max_value - value)` promove a maximização do valor da mochila.

5. Resultados

Após a execução do algoritmo, os resultados foram:

```
Parameters of the best solution : [1, 1, 0, 1, 0, 0]
Fitness value of the best solution = 0.05
Predicted output based on the best solution : 26
```

- **Melhor solução**: `[1, 0, 0, 1, 1, 0]`, indicando que os itens 1, 4 e 5 foram selecionados.
- **Valor total**: 26 dólares, que é o valor máximo obtido sem exceder a capacidade de 15 kg.

Análise e Justificativas

A escolha dos parâmetros foi feita visando um equilíbrio entre exploração (crossover uniforme e mutação) e exploração direcionada (seleção dos melhores pais via RWS). Utilizamos uma taxa de mutação relativamente baixa para evitar grandes mudanças nos indivíduos, mantendo a convergência do algoritmo.

A função fitness foi projetada para penalizar fortemente soluções que ultrapassem o limite de peso, mantendo apenas aquelas que são válidas. O uso de `1 / abs(max_value - value)` ajuda a evitar um fitness muito baixo para soluções sub-ótimas, favorecendo a maximização do valor.

6. Conclusão

A solução utilizando Algoritmo Genético apresentou resultados satisfatórios para o Problema da Mochila, maximizando o valor dentro da restrição de capacidade. A função fitness e os parâmetros foram adequadamente definidos para garantir uma boa exploração e evitar estagnação prematura. Para problemas maiores, ajustes nos parâmetros e no número de gerações podem ser necessários para obter resultados ainda melhores.