Algoritmos Genéticos combinatórios

Carolina Ribeiro Xavier

Setembro de 2022

1 Problema da Mochila binária

O problema da mochila binária consiste em um problema de maximização da utilidade dos itens levados (v), cada um dos n itens candidatos com peso armazenada em (p), restrito à capacidade da mochila(c).

Formalmente, dados dois vetores p e v de n posições e um número c, deseja-se encontrar um subconjunto X de $\{0, 1, , ..., n-1\}$ que maximize $v(X)$ sob a restrição $\sum p(X) \leq c$.

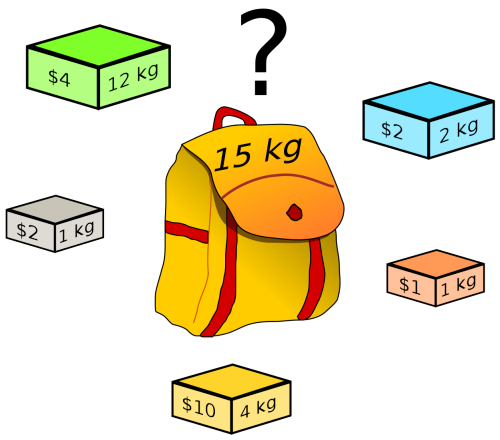

Figura 1: Ilustração do problema da Mochila

Diremos que $0, 1, 2, ..., n-1$ são os objetos do problema, $p[i]$ é o peso do objeto i, e que $v[i]$ é o valor(utilidade) do objeto i. Diremos que c é a capacidade da instância. Diremos ainda que um subconjunto X de $\{0, 1, 2, ..., n\}$ é uma mochila se $\sum p(X) \leq c$. O valor de uma mochila X é dado pela função $v(X)$, e nosso objetivo é encontrar uma mochila de valor máximo, e.g maximizar $v(X)$.

$$v(X) = \sum_{\forall i \in X} v[i] \tag{1}$$

2 Questões de projeto

- Função objetivo com **penalização**;
- Representação;
- Estratégia de seleção; ✓
- Cruzamento, pode ser feito o de n pontos; ✓
- Mutação, pode ser feita a de negação do bit; ✓
- Elitismo (usar o elitismo de 1); ✓

Valores como tamanho da população e número máximo de gerações do AG também devem ser testados para verificar o melhor cenário para solução do problema que se quer resolver.

2.1 Função objetivo

A função objetivo é uma função que avalia a proximidade do indivíduo da solução do problema, o problema tratado na primeira parte deste tutorial é o problema da

mochila binária, um problema de maximização sujeito à restrições. O valor de *fitness* ou aptidão do indivíduo será uma composição da utilidade da instância com uma penalização, caso a instância não seja uma solução viável (não é uma mochila).

Se a instância é viável, ou seja, o peso dos elementos em X não excedem a capacidade c, o fitneess é dado somente pela utilidade:

$$fitness = \sum_{\forall i \in X} v[i] \tag{2}$$

Caso contrário, faremos uma penalização.

$$fitness = \sum_{\forall i \in X} v[i] * (1 - (\sum_{\forall i \in X} p[i] - c)/c) \tag{3}$$

Observe que a segunda parte da função calcula o percentual de peso excedido, em seguida reduz a utilidade na proporção inversa. Este tipo de penalização é muito usado quando você pode ter alguma tolerância a violação das restrições.

Caso o seu AG esteja constantemente sendo dominado por soluções inviáveis, e isso não possa ser permitido, você pode usar uma penalização mais drástica, que não permitirá nunca que uma solução viável tenha *fitness* maior que uma não viável, pois tornará o *fitness* de todas as soluções inviáveis.

$$fitness = \sum_{\forall i \in X} v[i] - (\sum_{\forall i \in X} v[i] * (\sum p[i] - c)) \quad (4)$$

2.2 Representação

Para este problema temos duas alternativas:

- vetor binário indexado pelo id do objeto, onde o zero significa que o objeto não faz parte da solução e o 1 indica que ele faz parte da solução;
- vetor com os objetos presentes na solução, onde cada posição é um objeto. (este só deve ser usado se o número de objetos disponíveis for muito grande e

inviabilizar a opção anterior, pois teremos que admitir operações que varie o tamanho da solução e que não permita repetição de itens).

2.3 Estratégia de seleção

Você pode testar as duas formas de seleção que vimos até agora, e pode ainda fazer umas variações para ajustar o método que você escolheu.

Dicas para melhorias dos métodos:

- suavização da roleta (experimentem isso quando o mesmo pai está sendo selecionado várias vezes devido a uma grande diferença do *fitness* dele para os dos demais indivíduos, isso diminui a pressão seletiva, dando oportunidades a indivíduos com *fitness* piores, você pode usar o *rank*, por exemplo);

- torneio com o número de indivíduos maior (experimente quando o algoritmo estiver estagnado por muitas gerações sem aproximação do ótimo, em geral com a média de *fitness* bastante variável), isso aumenta a pressão seletiva;

2.4 Cruzamento

O cruzamento é uma operação muito importante para **intensificação** do espaço de busca, ele acontece entre dois indivíduos selecionados por algum critério.

O tipo cruzamento mais comum para representação por códigos binários é o cruzamento de n pontos. Nesse tipo de cruzamento são gerados, em geral, dois filhos para comporem a nova população. Um exemplo de cruzamento de n=2 pontos pode ser ilustrado pela Figura 2.

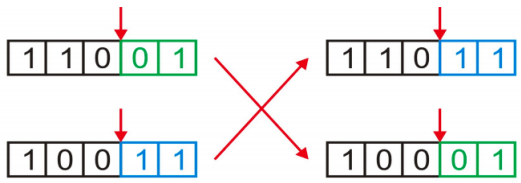

Figura 2: Cruzamento de um ponto

Este tipo de cruzamento pode ser usado nas duas representações, mas deve-se observar na segunda opção se não houve repetição de itens.

2.5 Mutação

A operação de mutação tem o papel de **diversificação** da população, muitas vezes os indivíduos ficam presos a ótimos locais e precisam de uma perturbação para que seja possível escapar desses locais.

Após a aplicação do operador de cruzamento, o operador de mutação é aplicado na população intermediária. Para cada indivíduo (por vezes para cada gene ou alelo do indivíduo) da população é sorteado um valor entre 0 e 1, caso esse valor seja menor ou igual a taxa de mutação estipulada o indivíduo (ou seu gene/alelo) é substituído. No caso da representação binária, pode-se manter a ideia anterior de inverter o bit, e no caso da representação por id de objetos pode-se substituir o objeto do cromossomo por outro aleatório, ou ainda remover(ou adicionar) um objeto aleatório.

Para isso é preciso fixar uma taxa de mutação e/ou alguns critérios de como ela pode variar durante a execução do algoritmo. Valores comuns para essa taxa não costumam ultrapassar 20%, sendo mais comuns valores de 1% a 10%.

Se o seu algoritmo está estagnado, você pode tentar após um número pré-fixado de gerações estagnadas, aumentar a taxa de mutação por algumas (poucas) gerações para dar diversidade à sua solução, depois retornar a taxa padrão.

3 Implementação

Agora vamos implementar um AG simples com representação à sua escolha para a mochila, e seguirá os passos do fluxograma da Figura 3.

Você deve definir a estrutura de dados que você irá armazenar os indivíduos e seus respectivos valores de *fitness*.

Use a alguma instância encontrada **aqui**.

Cada instância possui três arquivos, sendo:

pxx_c.txt, a capacidade da mochila;

pxx_w.txt, pesos dos objetos;

pxx_s.txt, valor ótimo para que você avalie a solução do seu AG.

- Tamanho da população;

- Número máximo de gerações a serem executadas;
- Critério de seleção de pais;
- Taxa de cruzamento;
- Taxa de mutação;
- Elitismo.

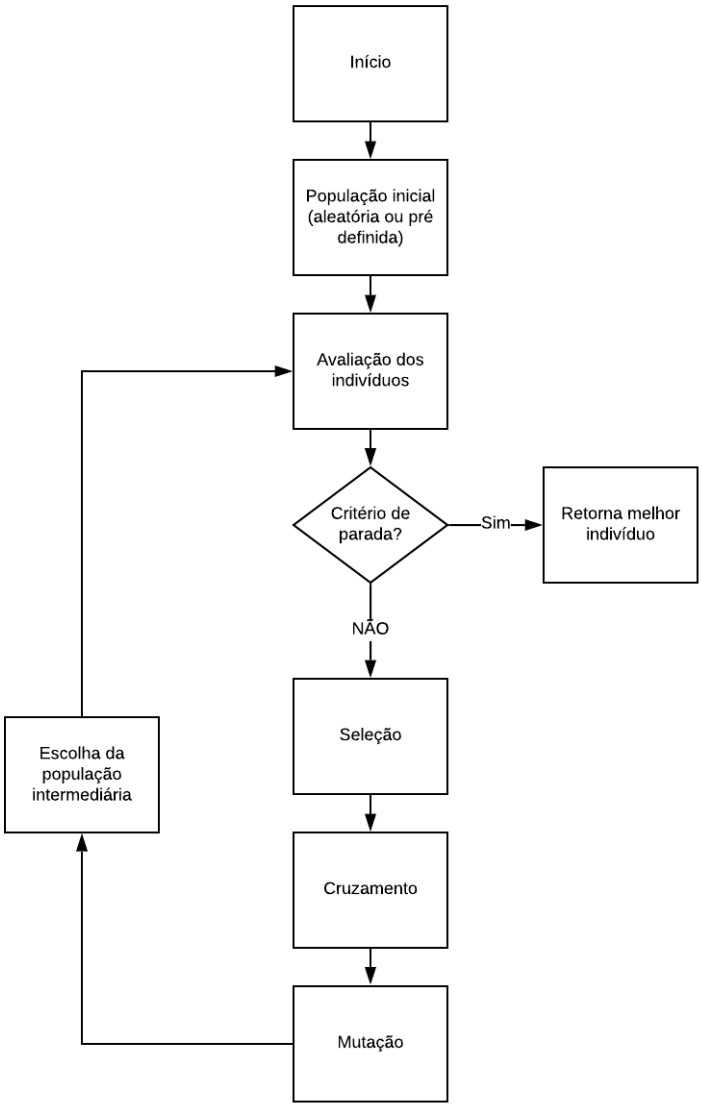

Figura 3: Fluxograma de um algoritmo genético básico