Fundamentos de Inteligência Artificial
Trabalho Prático II - Algoritmos Genéticos

Isabella Beatriz de Souza Gomes
Sarah Fernanda David Cunha
Vitor Hugo Lacerda Lana

Maio de 2025

Resumo

Este trabalho apresenta uma análise experimental da aplicação de Algoritmos Genéticos (AG) em dois contextos de otimização distintos: contínuo e discreto. Inicialmente, implementa-se um AG para minimizar uma função quadrática com 10 variáveis contínuas, investigando o impacto dos hiperparâmetros—tais como tamanho da população, taxas de cruzamento e mutação—sobre a qualidade das soluções e o desempenho computacional. Posteriormente, o algoritmo é adaptado para resolver o Problema da Mochila, uma clássica tarefa de otimização discreta. Neste segundo cenário, avalia-se especialmente como variações no tamanho da população e nas taxas de cruzamento e mutação afetam a eficiência e a proximidade das soluções encontradas em relação à solução ótima. Os resultados obtidos reforçam a importância crítica do ajuste adequado dos hiperparâmetros na utilização eficaz de Algoritmos Genéticos, proporcionando diretrizes para futuras aplicações desses métodos em problemas variados de otimização.

Sumário

1 Introdução

Algoritmos Genéticos (AG) são técnicas inspiradas na evolução biológica, especialmente na seleção natural, que têm se mostrado eficientes na resolução de problemas complexos de otimização. Esses algoritmos usam processos como seleção, cruzamento e mutação para buscar soluções próximas das ideais em problemas difíceis e amplos.

Neste trabalho, os AG são aplicados em dois tipos diferentes de problemas: um de otimização contínua e outro de otimização discreta. Primeiro, será abordada a minimização de uma função quadrática com 10 variáveis contínuas, avaliando como a escolha de parâmetros (tamanho da população, taxas de cruzamento e mutação) afeta o desempenho e as soluções encontradas.

Em seguida, será aplicado o AG ao Problema da Mochila, uma situação clássica de otimização discreta. Este problema consiste em escolher itens com valores e pesos definidos, visando maximizar o valor total sem ultrapassar a capacidade máxima da mochila. O Problema da Mochila é muito relevante por suas diversas aplicações práticas em áreas como logística e finanças.

O objetivo geral é analisar como diferentes ajustes nos parâmetros influenciam os resultados obtidos pelos Algoritmos Genéticos. Este trabalho está organizado em explicações teóricas básicas sobre AG, detalhes sobre as implementações específicas realizadas, resultados experimentais, discussões e, por fim, conclusões e sugestões para futuras pesquisas.

2 Tarefa I - Otimização

2.1 Definição do problema

Os algoritmos genéticos (AGs) são métodos de otimização e busca inspirados nos processos de seleção natural e evolução biológica, propostos inicialmente por John Holland na década de 1970 (Holland, 1975). Eles são especialmente eficientes para problemas nos quais métodos determinísticos apresentam dificuldades, como otimização de funções complexas, não-lineares e de muitos parâmetros. Assim,nessa tarefa desenvolverá a implementação e análise de um Algoritmo Genético aplicado à minimização de uma função quadrática de 10 variáveis contínuas. O objetivo principal do trabalho é investigar o comportamento do AG em um problema de otimização contínua, bem como compreender como os hiperparâmetros do algoritmo influenciam o desempenho e a qualidade das soluções obtidas.

2.2 Modelagem do Problema

A função objetivo a ser minimizada é a soma dos quadrados das variáveis:

$$\min f(x) = \sum_{i=1}^{10} x_i^2 \tag{1}$$

Esta função possui um mínimo global claro em $x = 0$, onde o valor mínimo da função é 0.

2.2.1 Variáveis de decisão

O problema envolve 10 variáveis contínuas x_i, cada uma restrita ao intervalo $[-5, 5]$. Essas variáveis representam a solução candidata a ser avaliada pelo algoritmo genético.

2.2.2 Restrições

As variáveis devem respeitar a restrição de domínio contínuo, ou seja, cada x_i deve permanecer no intervalo $[-5, 5]$ Essa restrição é tratada diretamente no algoritmo por meio da limitação dos valores após os operadores genéticos.

2.3 Implementação do algoritmo

2.3.1 Representação do Indivíduo (Cromossomo)

Cada indivíduo na população é representado por um vetor real de dimensão 10, onde cada gene corresponde a uma variável x_i. Essa codificação direta permite manipular diretamente os valores das variáveis contínuas.

2.3.2 Função de Aptidão (Fitness Function)

A aptidão do indivíduo é definida pelo valor da função objetivo. Como o problema é de minimização, o valor da função serve diretamente como métrica, sendo que menores valores indicam melhores soluções. Para fins computacionais, é comum inverter a função de aptidão para que valores maiores representam soluções melhores. Contudo, para simplicidade e clareza, neste trabalho utilizou-se a função objetivo diretamente, com foco na minimização.

2.3.3 Operadores Genéticos

Os operadores genéticos são os mecanismos que guiam a evolução da população e a busca por soluções ótimas. Neste trabalho, os principais operadores implementados são:

1. **Seleção (Pop)**
   A seleção é o processo de escolha dos indivíduos que irão gerar descendentes para a próxima geração. Para este problema de minimização, foi utilizado o método de seleção por torneio com tamanho fixo k=4. Nesse método, quatro indivíduos são selecionados aleatoriamente da população e o indivíduo com o menor valor da função objetivo (melhor aptidão) é escolhido como pai.Esse mecanismo assegura que soluções com menor valor da função objetivo têm maior probabilidade de reprodução, direcionando a população progressivamente para regiões com valores menores da função, ou seja, melhores soluções.

2. **Cruzamento (Cruz)**
   Após a seleção, os pais são combinados para gerar novos indivíduos por meio do crossover aritmético. Esse operador cria descendentes ao realizar combinações lineares entre os genes dos pais, promovendo diversidade e exploração do espaço de busca. A taxa de cruzamento define a probabilidade de aplicação do crossover para cada par selecionado.

3. **Mutação (Mut)**
   A mutação é aplicada a cada gene com uma pequena probabilidade e consiste na adição de um ruído aleatório retirado de uma distribuição normal com média zero. Isso introduz variações nos indivíduos, ajudando a manter a diversidade genética e a evitar a convergência prematura em mínimos locais. Após a mutação, os valores são limitados para permanecerem dentro do intervalo permitido $[-5, 5]$.

2.4 Configuração Experimental

2.4.1 Parâmetros do Algoritmo

Para avaliar o impacto dos hiperparâmetros no desempenho do AG, foram testadas diferentes configurações, variando:

- Taxa de cruzamento: $0.6, 0.8, 0.9$
- Taxa de mutação: $0.01, 0.04$
- Tamanho do torneio na seleção por torneio (k) : $(2, 3, 5)$
- População: $15, 25$

Cada combinação de parâmetros foi executada 2 vezes para garantir robustez estatística.

2.4.2 Critérios de Parada

O algoritmo é executado por um número fixo de gerações (20), pois o objetivo principal é observar o comportamento do AG dentro desse limite.

2.4.3 Métricas de Avaliação

Foram consideradas as seguintes métricas para comparação dos resultados:

- Melhor valor final encontrado: menor valor da função objetivo após 20 gerações
- Média do melhor valor entre execuções: média dos valores mínimos obtidos em 2 execuções
- Desvio padrão do melhor valor: para avaliar a consistência dos resultados
- Tempo médio de execução: tempo computacional médio para cada configuração

2.5 Resultados e Discussão

A tabela 1 apresenta os resultados obtidos para diferentes configurações dos hiperparâmetros do algoritmo genético (tamanho da população, taxa de cruzamento, taxa de mutação e tamanho do torneio de seleção). Foram avaliados o melhor valor da função objetivo alcançado, a média, o desvio padrão, o tempo médio de execução e os valores obtidos em cada execução.

Configuração	**Melhor Valor**	**Média**	**Desvio Padrão**	**Tempo Médio (s)**	**Valores por Execução**
Pop=15, Cruz=0.6, Mut=0.01, k=1	29.55	35.70	6.15	0.07	41.85, 29.55
Pop=15, Cruz=0.6, Mut=0.01, k=3	6.37	09.09	2.71	0.04	6.37, 11.80
Pop=15, Cruz=0.6, Mut=0.04, k=1	36.14	38.89	2.75	0.04	36.14, 41.64
Pop=15, Cruz=0.6, Mut=0.04, k=3	0.89	02.04	1.15	0.06	0.89, 3.19

Configuração	**Melhor Valor**	**Média**	**Desvio Padrão**	**Tempo Médio (s)**	**Valores por Execução**
Pop=15, Cruz=0.8, Mut=0.01, k=1	31.39	31.92	0.52	0.04	31.39, 32.44
Pop=15, Cruz=0.8, Mut=0.01, k=3	08.07	9.33	1.26	0.04	8.07, 10.59
Pop=15, Cruz=0.8, Mut=0.04, k=1	29.93	30.70	0.77	0.05	29.93, 31.47
Pop=15, Cruz=0.8, Mut=0.04, k=3	2.45	5.78	3.34	0.05	9.12, 2.45
Pop=15, Cruz=0.9, Mut=0.01, k=1	50.74	55.26	4.52	0.04	50.74, 59.78
Pop=15, Cruz=0.9, Mut=0.01, k=3	3.94	9.66	5.72	0.04	3.94, 15.38
Pop=15, Cruz=0.9, Mut=0.04, k=1	11.14	19.59	8.44	0.05	11.14, 28.03
Pop=15, Cruz=0.9, Mut=0.04, k=3	1.19	2.44	1.25	0.06	3.68, 1.19
Pop=25, Cruz=0.6, Mut=0.01, k=1	12.99	22.55	9.57	0.07	32.12, 12.99
Pop=25, Cruz=0.6, Mut=0.01, k=3	1.12	1.34	0.23	0.08	1.12, 1.57
Pop=25, Cruz=0.6, Mut=0.04, k=1	8.58	21.26	12.68	0.07	33.94, 8.58
Pop=25, Cruz=0.6, Mut=0.04, k=3	1.91	3.49	1.58	0.08	5.06, 1.91
Pop=25, Cruz=0.8, Mut=0.01, k=1	6.40	23.40	17.00	0.06	6.40, 40.40
Pop=25, Cruz=0.8, Mut=0.01, k=3	2.95	5.68	2.73	0.08	8.41, 2.95
Pop=25, Cruz=0.8, Mut=0.04, k=1	17.93	21.23	3.30	0.08	17.93, 24.53
Pop=25, Cruz=0.8, Mut=0.04, k=3	1.34	1.40	0.06	0.07	1.46, 1.34
Pop=25, Cruz=0.9, Mut=0.01, k=1	1.49	6.73	5.24	0.08	1.49, 11.97
Pop=25, Cruz=0.9, Mut=0.01, k=3	0.93	01.01	0.09	0.07	1.10, 0.93
Pop=25, Cruz=0.9, Mut=0.04, k=1	35.09	36.28	1.20	0.08	35.09, 37.48
Pop=25, Cruz=0.9, Mut=0.04, k=3	0.27	0.51	0.24	0.07	0.27, 0.75

Tabela 1: Tabela de resultados obtidos para diferentes configurações de hiperparâmetros

De forma geral, observa-se que:

- Tamanho da população (Pop): Populações maiores (25 indivíduos) tendem a alcançar

melhores resultados, com valores mínimos da função objetivo mais baixos em comparação a populações menores (15 indivíduos). Isso indica que uma maior diversidade inicial favorece a exploração do espaço de soluções.

- Taxa de cruzamento (Cruz): Taxas de cruzamento mais altas, especialmente 0.9, parecem beneficiar a convergência para melhores soluções em conjunto com alta taxa de mutação e torneio maior. Porém, taxas mais baixas como 0.6 ainda podem apresentar bons resultados quando combinadas com parâmetros adequados.

- Taxa de mutação (Mut): Taxas maiores (0.04) em geral melhoram a capacidade do algoritmo escapar de ótimos locais, contribuindo para resultados melhores e mais consistentes, principalmente quando combinadas com torneio de tamanho 3.

- Tamanho do torneio (k): Torneios maiores ($k = 3$) promovem maior pressão seletiva, e isso fica claro nos resultados, as melhores soluções (menor valor da função objetivo) geralmente foram obtidas com $k = 3$. Isso indica que uma seleção mais rigorosa dos pais favorece a qualidade da população.

De modo geral, os resultados mostram que a combinação dos parâmetros do algoritmo genético impacta bastante o desempenho e a qualidade das soluções encontradas. A pressão seletiva, representada pelo tamanho do torneio (k), é um fator crucial, torneios maiores ($k = 3$) tendem a melhorar significativamente a qualidade dos resultados, pois selecionam com mais rigor os indivíduos melhores para reprodução, acelerando a convergência do algoritmo.

Além disso, taxas mais altas de mutação e cruzamento favorecem a diversidade genética na população, ajudando o algoritmo a escapar de ótimos locais e explorando melhor o espaço de soluções. Em particular, a combinação de mutação alta (0.04) e cruzamento alto (0.8 ou 0.9) junto com torneio maior se destaca por apresentar os melhores resultados, com menor desvio padrão, indicando soluções mais estáveis e próximas do ótimo.

A população também tem papel importante: aumentar o número de indivíduos de 15 para 25 tende a melhorar a capacidade de busca do algoritmo, fornecendo mais diversidade inicial e permitindo uma exploração mais eficaz do espaço de soluções. Isso fica evidente nas melhores configurações que utilizam população maior, junto com taxas altas de mutação e cruzamento e torneio grande, que apresentam os melhores desempenhos.

Por outro lado, configurações com torneio pequeno ($k = 1$), baixa mutação e cruzamento, e população menor, resultam em pior desempenho, com soluções menos consistentes e mais distantes do ótimo, devido à baixa pressão seletiva e menor diversidade genética, que limitam a capacidade do algoritmo de evoluir adequadamente.

Em resumo, para obter melhores resultados, é ideal usar um torneio maior, taxas moderadas a altas de mutação e cruzamento, e uma população razoavelmente grande, garantindo um equilíbrio entre exploração e exploração no processo evolutivo. Essa combinação promove soluções de qualidade, com boa estabilidade e convergência eficiente.

2.5.1 Análise de sensibilidade dos hiperparâmetros

A análise de sensibilidade dos hiperparâmetros revela que cada um exerce influência significativa e distinta sobre o desempenho do algoritmo genético. O tamanho da população afeta diretamente a diversidade inicial da amostra: populações maiores tendem a proporcionar uma exploração mais ampla do espaço de soluções, o que geralmente resulta em soluções de melhor qualidade, porém com maior custo computacional.

A taxa de cruzamento é essencial para a recombinação de características entre indivíduos; taxas mais altas geralmente aumentam a capacidade do algoritmo de gerar soluções inovadoras, acelerando a convergência. Já a taxa de mutação tem papel crucial na manutenção da diversidade genética, evitando o aprisionamento em ótimos locais; taxas muito baixas podem levar a estagnação, enquanto taxas muito altas podem prejudicar a convergência ao introduzir muita aleatoriedade.

Por fim, o tamanho do torneio na seleção determina a pressão seletiva: torneios maiores aumentam a chance de escolher indivíduos mais aptos, promovendo uma evolução mais rápida e focada, mas podem reduzir a diversidade e aumentar o risco de convergência prematura. Assim, o equilíbrio entre esses hiperparâmetros é fundamental para o desempenho do algoritmo, sendo recomendado ajustar cada um com base no problema específico para garantir uma boa exploração e exploração do espaço de busca.

Assim, nota-se que:

- Tamanho da População (pop_size)
  A população inicial é criada com pop_size indivíduos. Populações maiores, como 25, mostraram melhor capacidade de encontrar soluções próximas do ótimo em algumas configurações, pois oferecem mais diversidade genética para explorar o espaço de soluções. No entanto, como o tempo médio por execução também aumentou, isso indica que a maior diversidade vem com custo computacional adicional. Populações menores (15) apresentaram resultados mais variáveis, às vezes convergindo rápido, mas com risco maior de ficar presas em soluções menos ótimas.

- Taxa de Cruzamento (taxa_c)
  A taxa de cruzamento no algoritmo controla com que frequência ocorre a recombinação dos genes entre pares de pais. Taxas intermediárias, como 0.6 e 0.8, equilibraram bem a exploração e a exploração, enquanto taxas muito altas (0.9) mostraram maior variabilidade nos resultados, com alguns casos apresentando valores ótimos, mas outros ficando presos em soluções piores. Isso sugere que a recombinação frequente nem sempre garante evolução mais estável no seu problema.

- Taxa de Mutação (taxa_m)
  A mutação no código altera aleatoriamente os genes dos indivíduos para manter a diversidade. Taxas de mutação mais baixas (0.01) às vezes limitaram essa diversidade, levando a resultados mais estáveis, porém menos variados. Já taxas maiores (0.04) ajudaram a escapar de mínimos locais em alguns casos, como no torneio com $k = 3$, onde melhores mínimos foram encontrados. Mas em algumas situações, mutações excessivas podem causar "ruído" e dificultar a convergência.

- Tamanho do Torneio (k)
  No processo de seleção via torneio, o parâmetro k define quantos indivíduos competem para serem selecionados como pais. Valores maiores ($k = 3$) aumentaram a pressão seletiva, levando a soluções melhores e mais consistentes, como mostrado pelos valores mais baixos da função objetivo e menores desvios padrão em algumas configurações. Já valores menores ($k = 1$) favoreceram maior diversidade, mas com desempenho médio inferior e mais dispersão nos resultados.

O gráfico 1 da evolução da função objetivo por geração mostra claramente como o algoritmo genético melhora as soluções ao longo das gerações. Inicialmente, observa-se uma grande

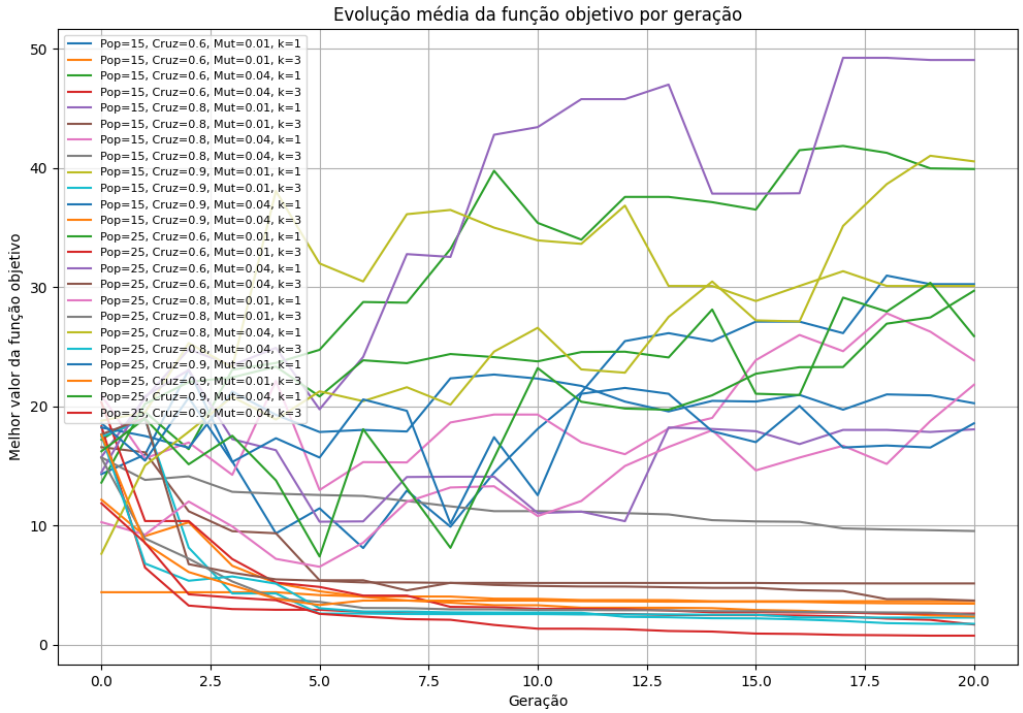

Figura 1: Gráfico de análise de resultados

oscilação nos valores da função objetivo, refletindo a grande diversidade da população inicial e a exploração intensa do espaço de busca. Essa *"ruidez"* ou variação acentuada nos primeiros passos é comum, pois o algoritmo ainda está *"descobrindo"* regiões promissoras da solução.

À medida que o número de gerações aumenta, a curva se estabiliza e tende a decrescer, indicando que as soluções estão se aprimorando e o algoritmo está convergindo para valores menores da função objetivo — no seu caso, minimizar a soma dos quadrados. Essa convergência mais suave nas gerações finais demonstra que o algoritmo está explorando regiões próximas do ótimo e as mutações e cruzamentos produzem melhorias mais refinadas.

Além disso, a taxa de queda e o formato do gráfico podem variar conforme os hiperparâmetros, como tamanho da população, taxa de mutação e cruzamento, e parâmetro de seleção (k). Configurações que promovem maior diversidade (exemplo: mutação mais alta, maior população) tendem a apresentar uma curva mais suave e maior capacidade de escapar de ótimos locais, enquanto configurações mais agressivas podem convergir rapidamente, mas com risco de estagnação.

Em resumo, o gráfico 1 evidencia o comportamento esperado do algoritmo genético: um rápido progresso inicial com ajustes finos conforme as gerações avançam, refletindo a exploração-exploração equilibrada do método.

2.6 Conclusão

A análise dos resultados obtidos por meio da aplicação do Algoritmo Genético ao problema de minimização demonstrou a relevância significativa dos hiperparâmetros para o desempenho do método. Entre os parâmetros avaliados, o tamanho da população revelou-se um dos mais influentes na qualidade e estabilidade das soluções. Populações maiores, especificamente com 25 indivíduos, proporcionaram melhor capacidade de exploração do espaço de busca, resultando em soluções mais consistentes, com menor desvio padrão e melhores valores médios. Essa

configuração contribuiu para evitar a convergência prematura em mínimos locais, problema comum em algoritmos evolutivos.

No que diz respeito à taxa de mutação, foi possível observar que valores mais elevados (por exemplo, 0.04) favoreceram a manutenção da diversidade genética ao longo das gerações. Isso evitou o fenômeno de estagnação, permitindo ao algoritmo escapar de regiões subótimas e alcançar melhores mínimos. Por outro lado, taxas de mutação muito baixas limitaram essa diversidade, o que resultou em soluções menos eficientes e maior variabilidade nos resultados.

A taxa de cruzamento também desempenhou papel fundamental no desempenho do algoritmo. Taxas mais altas, entre 0.8 e 0.9, mostraram-se mais eficazes na recombinação dos indivíduos, acelerando a convergência do algoritmo sem comprometer a diversidade da população. Essa recombinação eficiente das características das soluções permitiu uma melhor exploração do espaço de busca e otimização da função objetivo.

Adicionalmente, o parâmetro de seleção por torneio, representado pelo valor k, revelou impacto importante na pressão seletiva exercida sobre a população. O valor $k = 3$ proporcionou um equilíbrio adequado entre exploração e exploração, permitindo uma seleção eficiente dos melhores indivíduos sem sacrificar a diversidade necessária para evitar mínimos locais. Em contraste, valores menores de k resultaram em menor pressão seletiva e, consequentemente, em uma evolução mais lenta.

Com base nas evidências obtidas, a configuração que melhor atendeu aos objetivos do problema foi aquela que combinou uma população de 25 indivíduos, taxa de mutação de 0.04, taxa de cruzamento entre 0.8 e 0.9 e seleção por torneio com $k = 3$. Essa configuração garantiu o melhor compromisso entre qualidade da solução, estabilidade dos resultados e eficiência computacional, conforme corroborado pela análise da evolução da função objetivo ao longo das gerações.

Portanto, recomenda-se a utilização dessa combinação de hiperparâmetros para problemas similares de minimização usando Algoritmos Genéticos, especialmente quando se busca um desempenho robusto e confiável, que equilibre exploração e exploração do espaço de soluções.

3 Tarefa II - O problema da Mochila

3.1 Definição do problema

O Problema da Mochila é um problema clássico de otimização combinatória. Dado um conjunto de itens, cada um com um peso e um valor, o objetivo é determinar quais itens incluir em uma coleção de forma que o valor total seja o maior possível, sem que o peso total exceda uma capacidade limite (a capacidade da mochila).

Esta tarefa detalha a implementação de um Algoritmo Genético (AG) para resolver uma instância do Problema da Mochila. O objetivo é, além de encontrar uma solução para o problema, obter uma noção de como o tamanho da população e os hiperparâmetros, como taxa de cruzamento e mutação, afetam o desempenho do algoritmo.

3.2 Modelagem

Consideramos um conjunto de n itens, onde cada item i possui um peso w_i e um valor p_i. A mochila tem uma capacidade máxima de carga C. O objetivo é selecionar um subconjunto de itens tal que o valor total dos itens selecionados seja maximizado, sem que o peso total exceda a capacidade C.

Para modelar a decisão de incluir ou não um item, definimos uma variável binária x_i:

- $x_i = 1$, se o item s for selecionado para ser carregado na mochila.
- $x_i = 0$, se o item i não for selecionado.

A função objetivo visa maximizar a soma dos valores dos itens selecionados:

$$\max \sum_{i=1}^{n} p_i x_i \tag{2}$$

3.2.1 Restrição

A restrição garante que a soma dos pesos dos itens selecionados não exceda a capacidade máxima da mochila:

$$\sum_{i=1}^{n} p_i x_i \leq C \tag{3}$$

3.2.2 Variáveis de decisão

$$x_i \in \{0, 1\} \quad \text{para } i = 1, \ldots, n \tag{4}$$

3.3 Implementação do Algoritmo Genético

3.3.1 Representação do Indivíduo (Cromossomo)

Cada indivíduo na população representa uma solução candidata e é codificado como um vetor binário de tamanho N (o número total de itens). Cada gene x_i no cromossomo indica se o item i está incluído na mochila (1) ou não (0).

3.3.2 Função de Aptidão (Fitness Function)

A aptidão de um indivíduo é calculada da seguinte forma:

- Calcula-se o valor total V e o peso total W dos itens selecionados pelo indivíduo.
- Se $W \leq C$ (a restrição de capacidade é satisfeita), então a aptidão é V.
- Se $W > C$ (a restrição é violada), o indivíduo é penalizado.

Na implementação feita, a aptidão é definida como 0 para indivíduos inviáveis. Essa abordagem de penalização na função objetivo é uma forma de lidar com restrições.

3.3.3 Seleção

Foi implementado o método de seleção por torneio. Nesta estratégia, k indivíduos são selecionados aleatoriamente da população atual, e o indivíduo com a maior aptidão entre eles é escolhido como um dos pais para a próxima geração. Esta escolha foi feita por ser um método robusto e comumente utilizado, que permite ajustar a pressão seletiva através do tamanho do torneio k.

3.4 Operadores Genéticos

- Cruzamento (Crossover)
  Utilizou-se o operador de cruzamento de um ponto. Dois indivíduos pais são selecionados. Se um número aleatório for menor que a taxa de cruzamento pré-definida, um ponto de corte é escolhido aleatoriamente ao longo do comprimento do cromossomo. Dois novos descendentes (filhos) são gerados trocando os segmentos dos pais a partir do ponto de corte. Se o cruzamento não ocorrer, os filhos são cópias dos pais.

- Mutação
  Para este problema combinatório com representação binária, o operador de mutação implementado foi o *bit-flip*. Cada gene (*bit*) de um indivíduo tem uma pequena probabilidade (taxa de mutação por gene) de ser invertido (0 para 1 ou 1 para 0). A mutação introduz diversidade na população, ajudando a evitar a convergência prematura.

3.5 Configuração Experimental

3.5.1 Instância do Problema

Os experimentos foram realizados utilizando os dados fornecidos no arquivo mochila.txt contemplando os seguintes parâmetros:

- Número de Itens: 50
- Capacidade da Mochila: 14239

3.5.2 Hiperparâmetros Testados

Para avaliar o impacto dos hiperparâmetros no desempenho do AG, foram variados sistematicamente:

- Tamanho da População: [20, 50, 100, 200]
- Taxa de Cruzamento: [0.6, 0.7, 0.8, 0.85, 0.9, 0.95]
- Taxa de Mutação por Gene: [0.001, 0.01, 0.05, 0.1]

3.6 Métricas de Avaliação

Para comparar os resultados das diferentes configurações, as seguintes métricas foram consideradas:

- Melhor Valor de Aptidão Encontrado: O valor máximo da função objetivo alcançado pela melhor solução ao final das execuções.
- Média do Melhor Valor de Aptidão: Cada configuração foi executada 10 vezes, e a média dos melhores valores encontrados foi registrada.
- Desvio Padrão do Melhor Valor de Aptidão: Para avaliar a consistência dos resultados.
- Tempo Médio de Execução: O tempo computacional médio para cada configuração.

3.7 Resultado e Discussão

Os resultados dos experimentos são resumidos nas tabelas e gráficos abaixo.

3.7.1 Impacto do Tamanho da População

Foram testados tamanhos de população de 20, 50, 100 e 200 indivíduos, mantendo os demais parâmetros constantes. Para cada configuração, o algoritmo foi executado 10 vezes, com os resultados consolidados na tabela 2.

Os dados indicam que populações maiores tendem a produzir soluções de maior qualidade, ou seja, com maior valor total dos itens selecionados dentro do limite de capacidade da mochila. A melhor solução foi registrada com uma população de 200 indivíduos, alcançando um valor total de 15.533. Além disso, populações maiores também apresentaram médias mais elevadas e menor variação entre as execuções, o que sugere maior estabilidade e consistência nos resultados.

No entanto, observou-se um aumento no tempo médio de execução conforme o crescimento da população, variando de 0,0250 segundos (população de 20) até 0,2519 segundos (população de 200). Esse comportamento é esperado, pois populações menores tendem a convergir mais rapidamente, mas correm maior risco de se prender em ótimos locais devido à baixa diversidade. Em contraste, populações maiores exploram mais amplamente o espaço de busca, aumentando as chances de encontrar soluções superiores, embora com maior custo computacional.

Dessa forma, conclui-se que, apesar do aumento no tempo de execução, o uso de populações maiores favorece tanto a qualidade quanto a robustez das soluções obtidas pelo algoritmo genético.

Tamanho da População	**Melhor Valor**	**Média dos Melhores Valores**	**Desvio Padrão**	**Tempo Médio**
20	15549	15393.30	105.51	0.0250
50	15598	15460.30	80.77	0.0628
100	15511	15436.30	54.83	0.1282
200	15533	15455.40	53.18	0.2519

Tabela 2: Desempenho do Algoritmo Genético para diferentes tamanhos de população no problema da mochila.

3.7.2 Impacto da Taxa de Cruzamento

Nesta etapa, foram avaliadas diferentes taxas de cruzamento no Algoritmo Genético para o problema da mochila, com tamanho de população fixado em 100 indivíduos. As taxas testadas foram: 0,60; 0,70; 0,80; 0,85; 0,90 e 0,95. Para cada configuração, o algoritmo foi executado 10 vezes, e os resultados médios estão apresentados na Tabela 3.

Dentre os valores testados, a taxa de cruzamento de 0,85 obteve o melhor desempenho médio (15476,90), além de um bom equilíbrio entre qualidade e variabilidade das soluções. Embora a maior solução isolada tenha ocorrido com taxa 0,60 (15655), essa configuração apresentou maior desvio padrão (77,73), indicando menos estabilidade nos resultados. Da mesma forma, a taxa 0,95 também obteve um valor máximo elevado (15624), mas com desvio padrão elevado (86,98), o que aponta para menor consistência.

Quanto ao tempo de execução, a taxa de cruzamento 0,70 apresentou o maior tempo médio (0,2049 segundos), o que indica maior custo computacional nessa configuração. As outras taxas mantiveram tempos médios próximos entre 0,12 e 0,16 segundos, mostrando que a taxa de cruzamento não afeta drasticamente o desempenho computacional.

Conclui-se que a taxa de cruzamento de 0,85 apresenta o melhor compromisso entre qualidade média das soluções, consistência dos resultados e tempo de execução. Assim, recomenda-se seu uso para o problema e parâmetros avaliados, embora as outras taxas também gerem resultados próximos em termos de desempenho.

Taxa de Cruzamento	**Melhor Valor Encontrado**	**Média dos Valores Totais**	**Desvio Padrão**	**Tempo Médio (s)**
0,60	15655	15460,90	77,73	0,1272
0,70	15517	15433,30	54,89	0,2049
0,80	15508	15427,60	62,03	0,1618
0,85	15592	15476,90	69,47	0,1258
0,90	15574	15435,00	68,99	0,1257
0,95	15624	15460,10	86,98	0,1247

Tabela 3: Desempenho do Algoritmo Genético para diferentes taxas de cruzamento no problema da mochila.

3.7.3 Impacto da Taxa de Mutação

Foram realizados experimentos com quatro taxas de mutação: 0.001, 0.01, 0.05 e 0.1, com o objetivo de avaliar como esse parâmetro influencia o desempenho do Algoritmo Genético na resolução do problema da mochila. Os resultados estão na Tabela 4.

A taxa de 0.01 apresentou os melhores resultados gerais, com o maior valor máximo encontrado (15653) e a maior média de valores totais (15555,50). Além disso, obteve um baixo desvio padrão (79,78), indicando boa consistência nas soluções, e foi também a configuração com o menor tempo médio de execução (0,1550 s). Esses resultados sugerem que essa taxa equilibra bem a introdução de diversidade genética sem comprometer a convergência do algoritmo.

A taxa de 0.001 teve o pior desempenho médio (15207,80), além de apresentar o maior desvio padrão (114,11), o que reflete maior variabilidade nas soluções e menor qualidade média. Isso indica que uma taxa de mutação muito baixa pode dificultar a exploração do espaço de soluções.

As taxas mais altas, 0.05 e 0.1, apresentaram médias intermediárias (15457,10 e 15284,70, respectivamente), com desvios padrões menores, o que aponta para soluções mais estáveis, porém com desempenho inferior à taxa de 0.01. Além disso, essas configurações também tiveram tempos de execução mais altos.

Em resumo, a taxa de mutação de 0.01 foi a mais eficiente, proporcionando o melhor equilíbrio entre qualidade das soluções, estabilidade e desempenho computacional.

Taxa de Mutação	Melhor Valor	Média dos Melhores Valores	Desvio Padrão	Tempo Médio
0.001	15435	15207.80	114.11	0.1988
0.01	15653	15555.50	79.78	0.1550
0.05	15550	15457.10	56.32	0.4235
0.1	15415	15284.70	60.86	0.3561

Tabela 4: Desempenho do Algoritmo Genético para diferentes taxa de mutação no problema da mochila.

3.8 Conclusão

A implementação do Algoritmo Genético para o Problema da Mochila permitiu avaliar de forma sistemática o impacto de diferentes hiperparâmetros no desempenho da busca por soluções ótimas. Os experimentos demonstraram que o aumento do tamanho da população melhora a qualidade e a consistência das soluções, ainda que com maior custo computacional.

A taxa de cruzamento ideal foi 0,85, proporcionando um bom equilíbrio entre desempenho e estabilidade. Já a taxa de mutação de 0,01 se destacou como a mais eficiente, oferecendo a melhor média de soluções com consistência e menor tempo de execução.

Conclui-se, portanto, que a escolha adequada dos parâmetros é essencial para a eficácia dos algoritmos evolutivos, e pequenas variações podem impactar os resultados obtidos.

4 Conclusão do trabalho

Com base nas análises realizadas nas duas tarefas propostas neste trabalho, pode-se concluir que os Algoritmos Genéticos são ferramentas versáteis e eficazes para a resolução de problemas de otimização, tanto em domínios contínuos quanto discretos. A aplicação dos AG à minimização de uma função quadrática com variáveis contínuas e ao Problema da Mochila evidenciou que o ajuste adequado dos hiperparâmetros é fundamental para o sucesso do algoritmo.

Observou-se que combinações específicas de parâmetros — como populações maiores, taxas de mutação moderadamente altas e taxas elevadas de cruzamento — promovem melhor desempenho, evitando estagnações e favorecendo a diversidade genética, o que, por sua vez, contribui para a obtenção de soluções mais próximas do ótimo global. O uso da seleção por torneio com valor $k = 3$ demonstrou ser eficiente em manter o equilíbrio entre exploração e exploração do espaço de busca, característica essencial para evitar a convergência prematura e maximizar o desempenho.

A partir das implementações realizadas, ficou claro que os Algoritmos Genéticos não apenas são capazes de lidar com diferentes naturezas de problemas, mas também se adaptam bem a contextos diversos, desde que os parâmetros sejam cuidadosamente calibrados. Dessa forma, este trabalho reforça a importância do estudo empírico e da experimentação no uso de AGs, bem como a necessidade de compreender profundamente o problema em questão para definir uma configuração eficiente.

5 Disponibilidade do Código

Todo o código desenvolvido e utilizado neste trabalho prático está disponível publicamente em um repositório GitHub. O acesso pode ser realizado através do seguinte endereço:

`https://github.com/vitorlana/fundamentos-ia/tree/main/tp1`

Neste repositório encontram-se todos os arquivos necessários para reprodução dos experimentos realizados, incluindo scripts, dados utilizados e documentação complementar.