**1.0 Genetic Algorithm**

**1.1 Taxonomia**

O Algoritmo Genético é uma Estratégia Adaptativa e uma técnica de Otimização Global. É um Algoritmo Evolutivo e pertence ao estudo mais amplo de Computação Evolucionária. O Algoritmo Genético é um irmão de outros algoritmos evolutivos, como programação genética (Seção 3.3), Estratégias de Evolução, Programação Evolutiva e Sistemas Classificadores de Aprendizagem.O Algoritmo Genético é pai de um grande número de técnicas variantes e subcampos numerosos demais para serem listados.

**1.2 Inspiração**

O Algoritmo Genético é inspirado na genética populacional (incluindo hereditariedade e frequências genéticas) e evolução em nível populacional, bem como a compreensão mendeliana da estrutura (como cromossomos, genes, alelos) e mecanismos (como recombinação e mutação). Esta é a chamada síntese nova ou moderna da evolução

biologia.

**1.3 Metáfora**

Os indivíduos de uma população contribuem com seu material genético (chamado de genótipo) proporcional à sua adequação ao seu genoma expresso (chamado de fenótipo) ao seu ambiente, na forma de descendentes. A próxima geração é criada através de um processo de acasalamento que envolve recombinação de genomas de dois indivíduos na população com o introdução de erros de cópia aleatórios (chamados de mutação). Este iterativo processo pode resultar em um melhor ajuste adaptativo entre os fenótipos dos indivíduos de uma população e do ambiente.

**1.4 Estratégia**

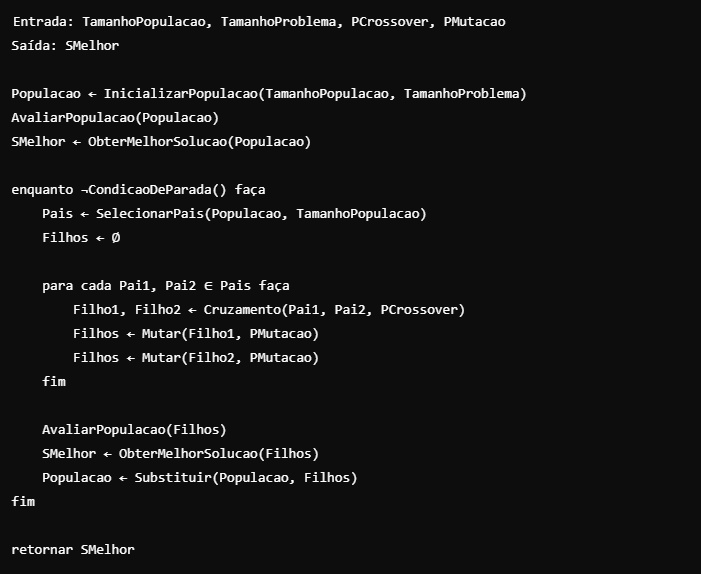
O objetivo do Algoritmo Genético é maximizar o retorno de soluções candidatas na população contra uma função de custo do domínio do problema. A estratégia do Algoritmo Genético é repetidamente empregar substitutos para os mecanismos genéticos de recombinação e mutação na população de soluções candidatas, onde a função de custo (também conhecida como função objetivo ou de aptidão) aplicada a uma representação decodificada de um candidato rege as contribuições probabilísticas de um determinado

**1.5 Procedimento**

O Algoritmo 1.1 fornece uma listagem em pseudocódigo do Algoritmo Genético

para minimizar uma função de custo.

**1.5.1 Pseudocódigo do Algoritmo Genético**



**1.6 Heurística**

* Strings binárias (chamadas de ‘bitstrings’) são a representação clássica, pois podem ser decodificadas para quase qualquer representação desejada. Variáveis ​​​​de valor real e inteiras podem ser decodificadas usando o binário método decimal codificado, métodos de complemento de um ou dois, ou o método do código cinza, o último dos quais é geralmente preferido.
* Representações específicas de problemas e operadores genéticos customizados devem ser adotadas, incorporando o máximo de informações prévias sobre possível o domínio do problema.
* O tamanho da população deve ser grande o suficiente para fornecer algoritmos Evolutivos cobertura suficiente do domínio e mistura dos subcomponentes úteis da solução.
* O Algoritmo Genético é classicamente configurado com uma alta probabilidade de recombinação (como 95%-99% da população selecionada) e uma baixa probabilidade de mutação (como 1 L onde L é o número de componentes em uma solução).
* A seleção proporcional à adequação de soluções candidatas para contribuir para a próxima geração não deve ser nem muito gananciosa (para evitar a aquisição de soluções candidatas mais adequadas) nem demasiado aleatórias.

**1.7 Explicação do código**

A Listagem 1.8 apresenta uma implementação do Algoritmo Genético na linguagem de programação Python. O problema em questão é a maximização de uma função objetivo, onde buscamos encontrar a melhor solução através da evolução de uma população de indivíduos representados por listas de inteiros. A função de fitness utilizada simplesmente calcula a soma dos valores de cada indivíduo, servindo como um critério para avaliar sua qualidade.

A implementação do Algoritmo Genético é feita de forma modular, onde as principais etapas incluem a inicialização da população, a avaliação de cada indivíduo, a seleção dos pais, o cruzamento para gerar novos filhos, a mutação dos filhos e a substituição da população atual pelos novos indivíduos.

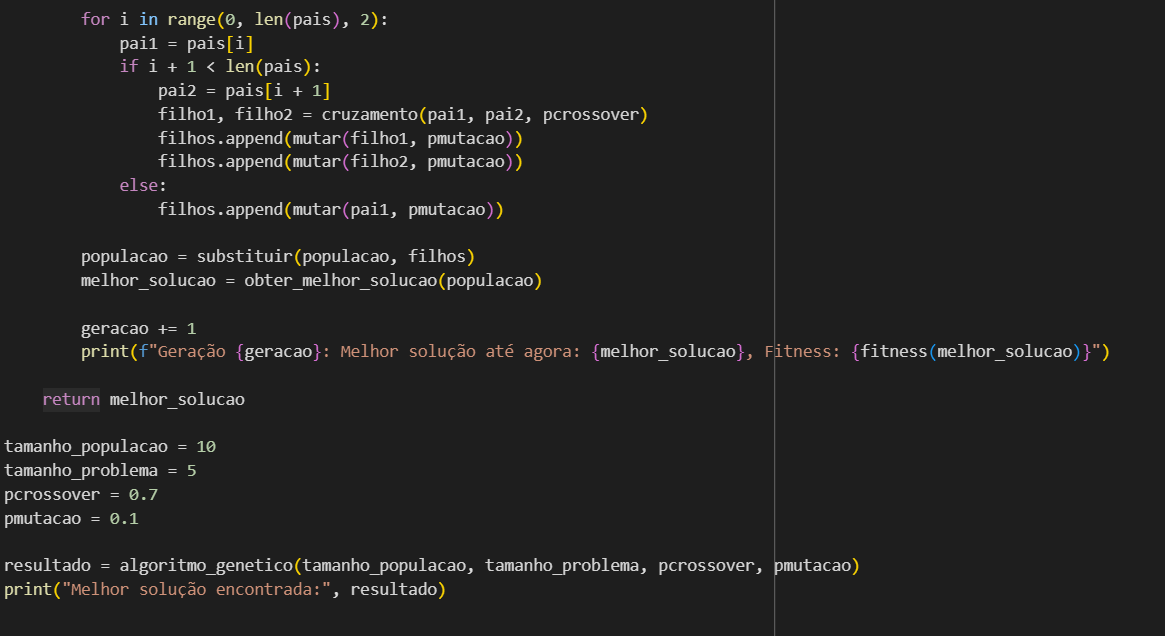
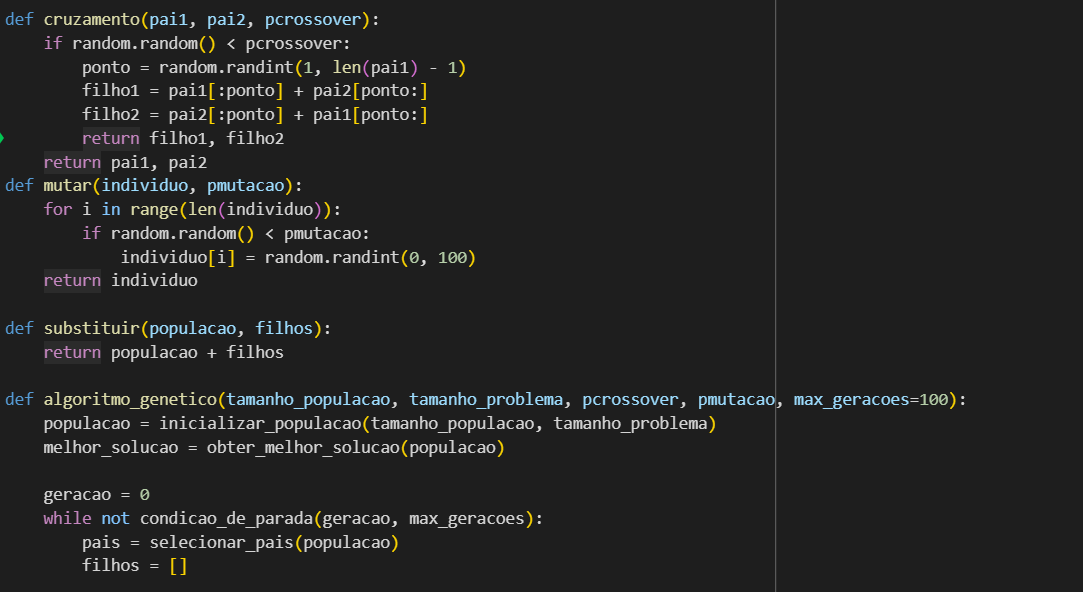
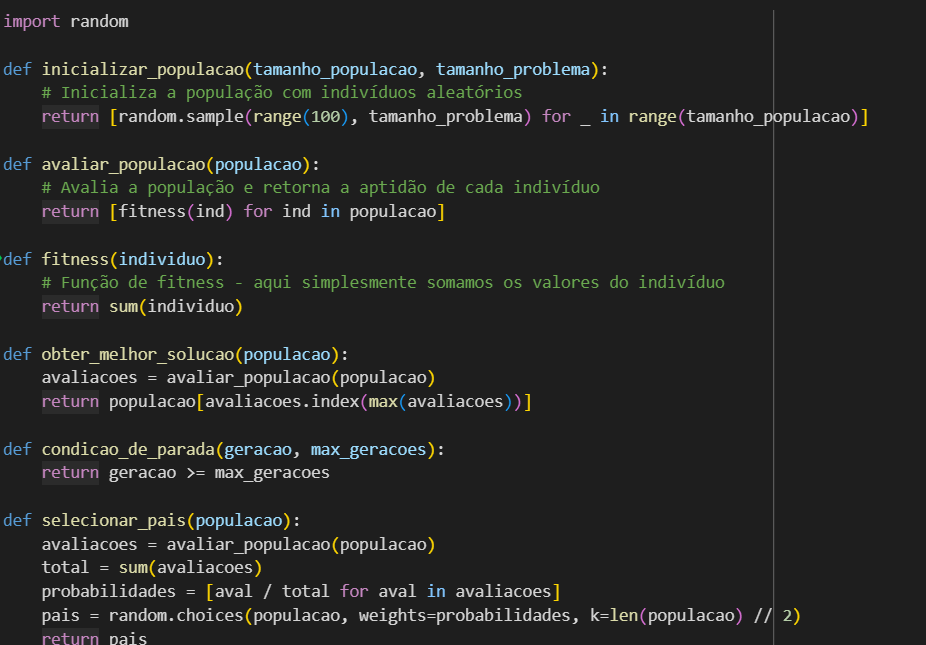
O processo começa com a função inicializar\_populacao, que gera uma lista de indivíduos aleatórios, cada uma consistindo de um número específico de elementos. Em seguida, a função avaliar\_populacao é chamada para calcular a aptidão de cada indivíduo, utilizando a função fitness, que soma os valores dos elementos da lista.

O Algoritmo também incorpora um mecanismo de seleção baseado na aptidão, onde os pais são escolhidos aleatoriamente, mas com uma probabilidade que favorece aqueles com maior aptidão. Isso é realizado pela função selecionar\_pais, que utiliza a seleção de roleta.

A parte do cruzamento é implementada na função cruzamento, onde dois pais geram dois filhos com base em uma probabilidade definida. O cruzamento é realizado em um ponto aleatório da lista. Já a mutação, realizada na função mutar, altera aleatoriamente alguns dos genes dos filhos, garantindo diversidade na população.

Após a geração dos filhos, a população antiga é substituída pelos novos indivíduos através da função substituir. O algoritmo continua a executar essas etapas até que uma condição de parada seja atendida, definida pelo número máximo de gerações.

Finalmente, a melhor solução encontrada durante o processo é retornada, permitindo a identificação do indivíduo com a maior aptidão ao longo das iterações.

**1.8 Código** **Funcional** 

Código disponível no [Github](https://github.com/MateusSales02/Algoritmos-Evolucion-rios)

**1.9 Referências**

**Fontes Primárias**

Holland é o avô da área que se tornou Algoritmos Genéticos. Holland investigou sistemas adaptativos no final da década de 1960 propondo uma formalismo de sistema adaptativo e estratégias adaptativas referidas como 'planos adaptativos'. O referencial teórico de Holland foi investigado e elaborado por seu Ph.D. estudantes da Universidade de Michigan. Rosenberg investigou um modelo químico e molecular de um biológico plano adaptativo inspirado. Bagley investigou metaambientes e um plano genético adaptativo denominado algoritmo genético aplicado a um jogo simples chamado hexapawn. Cavicchio elaborou ainda mais o plano genético adaptativo, propondo inúmeras variações, referindo-se a alguns como 'planos reprodutivos'. Outras contribuições importantes foram feitas por Frantz que investigou o que foi chamado de algoritmos genéticos para pesquisa, e Hollstien, que investigou planos genéticos para controle adaptativo e função otimização. De Jong realizou uma investigação seminal sobre o modelo genético adaptativo (planos genéticos) aplicado à função contínua otimização e seu conjunto de problemas de teste adotados ainda são comumente usado. Holland escreveu o livro seminal sobre sua pesquisa com foco no formalismo de sistemas adaptativos proposto, no sistema reprodutivo e planos adaptativos genéticos e forneceu uma estrutura teórica para o mecanismos utilizados e explicação para as capacidades do que seria tornam-se algoritmos genéticos.

**Saber mais**

O campo dos algoritmos genéticos é muito grande, resultando em grandes números de variações da técnica canônica. Goldberg fornece um clássico visão geral do campo em um artigo de revisão, assim como Mitchell. Whitley descreve um tutorial clássico para o Algoritmo Genético cobrindo ambos preocupações práticas e teóricas. O algoritmo é altamente modular e existe um subcampo para estudar cada subprocesso, especificamente: seleção, recombinação, mutação e representação. O Algoritmo Genético é mais comumente usado como técnica de otimização, embora também deva ser considerada uma técnica geral estratégia adaptativa. O teorema do esquema é uma explicação clássica pelo poder do Algoritmo Genético proposto por Holland, e investigado por Goldberg sob o nome de hipótese do bloco de construção.O livro clássico sobre algoritmos genéticos como uma otimização e A técnica de aprendizado de máquina foi escrita por Goldberg e fornece uma revisão aprofundada e um estudo prático da abordagem. Mitchell fornece um texto de referência contemporâneo que apresenta a técnica e o campo.

*Ícone "Verificada pela comunidade"*