SÉRGIO POLIMANTE SOUTO

PÓS TECH

IA PARA DEVS

INTRODUÇÃO AO ALGORITMO GENÉTICO

AULA 03

SUMÁRIO

O QUE VEM POR AÍ?

Nesta aula, aprofundaremos os princípios dos Algoritmos Genéticos. Revisitaremos o funcionamento do algoritmo, guiados por um diagrama que descreve cada passo. Exploraremos cada etapa, desde a codificação de soluções até os operadores genéticos, como seleção, crossover e mutação.

HANDS ON

Nesta videoaula, vamos começar revisando o funcionamento do algoritmo, relembrando a inspiração na natureza e na teoria da evolução por seleção natural.

SAIBA MAIS

A evolução por seleção natural, como descrita por Charles Darwin, e os algoritmos genéticos compartilham notáveis semelhanças em seus processos fundamentais. No contexto da natureza, as espécies herdam características genéticas dos pais por meio do cruzamento, enquanto nos algoritmos genéticos, soluções para um problema são representadas por indivíduos por meio de estruturas de dados, que passam suas características para a próxima geração por meio do operador de cruzamento, que produz uma nova estrutura de dados produto da combinação da estrutura de dados dos pais. A introdução de mutações proporciona a variedade necessária para explorar novas soluções. No ambiente natural, a pressão seletiva molda a sobrevivência das espécies, favorecendo os mais adaptados. De maneira análoga, em algoritmos genéticos, a seleção das melhores soluções é guiada por uma função de aptidão, que atua como uma espécie de pressão seletiva artificial. Os indivíduos mais aptos são preferencialmente escolhidos para reprodução, tendo maior chances de perpetuar suas características bem-sucedidas na próxima geração. Essa convergência para soluções mais eficientes espelha o processo evolutivo, onde cada nova geração carrega consigo um conjunto aprimorado de características, representando um paralelo fascinante entre a evolução natural e a computacional.

O diagrama a seguir descreve o processo do algoritmo genético e fornece uma visão clara do processo do início ao fim.

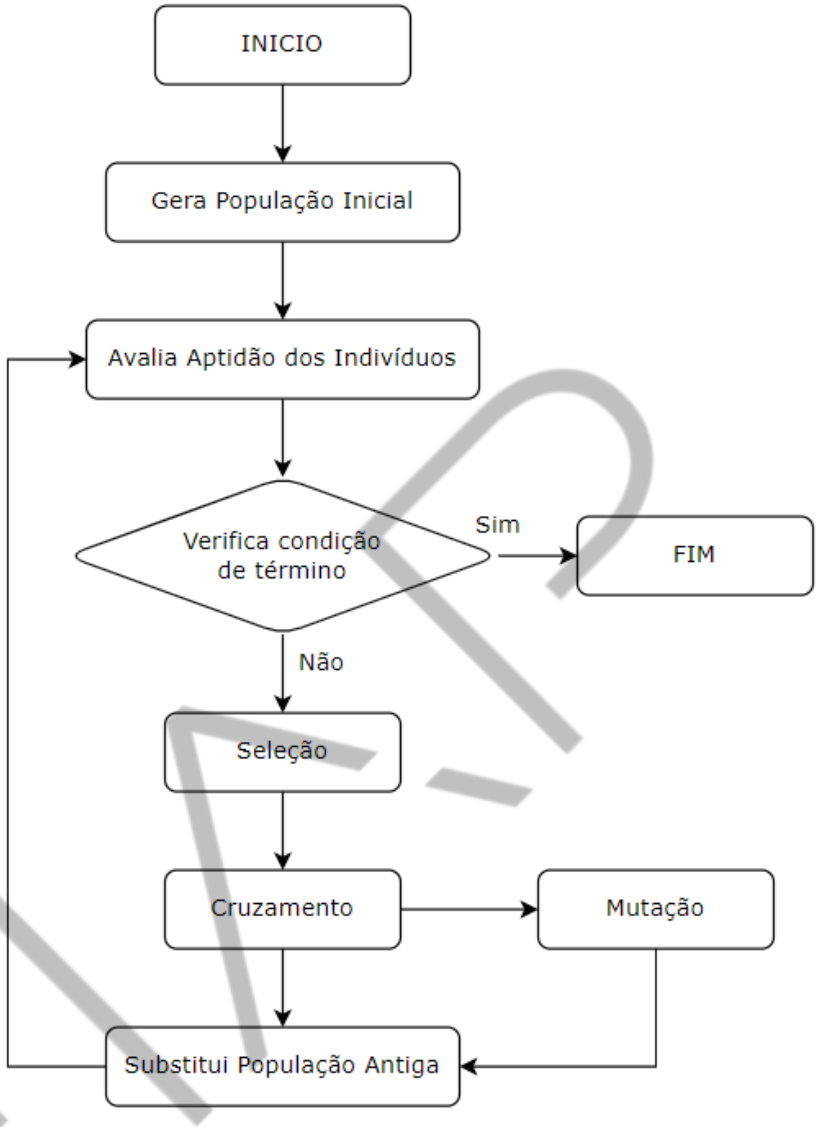

Figura 1 - Fluxograma Genérico do Algoritmo Genético
Fonte: elaborado pelo autor (2024), adaptado por FIAP (2024)

REPRESENTAÇÃO DOS INDIVÍDUOS (CODIFICAÇÃO)

Nos aprofundaremos em cada uma das etapas, começando pela representação de soluções por meio da codificação genética. Veremos diferentes tipos de codificação como binária, real, combinatória e hibrida. A codificação binária é a forma mais simples, onde cada característica da solução é representada por uma cadeia de bits, permitindo uma manipulação eficiente, mas requerendo uma interpretação adequada. Já a codificação real utiliza valores numéricos contínuos para representar parâmetros, sendo mais intuitiva para problemas que envolvem grandezas físicas ou variáveis contínuas. A codificação combinatória, por sua vez, é ideal para problemas de

otimização discreta, representando soluções como sequências de elementos. Por fim, a codificação híbrida combina diferentes abordagens, permitindo flexibilidade na representação conforme as características do problema. Essa diversidade de codificações oferece aos algoritmos genéticos uma adaptabilidade crucial para abordar uma ampla gama de desafios em otimização e busca.

Lembre-se: a codificação significa uma maneira de representarmos uma solução usando um conjunto de dados, que pode ser binário, real, combinatório ou híbrido. Vamos ver alguns exemplos de codificação de problemas reais para tornar nosso exemplo mais compreensível.

CODIFICAÇÃO BINÁRIA: PROGRAMAÇÃO DE HORÁRIOS EM INDÚSTRIAS

- **Problema:** alocar recursos humanos em diferentes turnos, considerando restrições de disponibilidade, preferências e requisitos de habilidades.
- **Codificação Binária:** cada gene representa a atribuição de um funcionário a um turno de trabalho, onde 1 indica que o funcionário está programado para esse turno e 0 indica que não está.
- **Exemplo:** para uma equipe de trabalhadores A, B e C, e três turnos de trabalho X, Y e Z, a solução [1X, 0Y, 1Z] indica que os trabalhadores A e C estão programados para os turnos X e Z, respectivamente.

CODIFICAÇÃO REAL: OTIMIZAÇÃO DE PROCESSOS QUÍMICOS EM UMA FÁBRICA

- **Problema:** encontrar os valores ideais para os parâmetros de um processo químico, considerando variáveis como temperatura, pressão e concentração.
- **Codificação Real:** cada variável é representada por um gene com valores reais, correspondendo a configurações específicas do processo.
- **Exemplo:** para otimizar um processo de síntese química, os genes [150°C, 3 atm, 0.2 M] podem representar a temperatura, pressão e concentração ideais.

CODIFICAÇÃO COMBINATÓRIA: ROTEIRIZAÇÃO DE VEÍCULOS PARA ENTREGA

- **Problema: determinar** as rotas mais eficientes para uma frota de veículos realizar entregas a diferentes destinos, minimizando o tempo e os custos.
- **Codificação Combinatória:** cada gene representa um destino, e a ordem dos genes representa a sequência de entrega para cada veículo.
- **Exemplo:** se temos três veículos V1, V2 e V3, e destinos A, B e C, a solução [V1: A, C, B; V2: B, A, C; V3: C, B, A] indica as rotas planejadas para cada veículo.

CODIFICAÇÃO HÍBRIDA: PROJETO DE SISTEMAS DE CONTROLE DE TRÁFEGO URBANO

- **Problema:** projetar a localização ideal de semáforos (representada por coordenadas reais) e decidir sobre os planos de sinalização (representados por genes binários).
- **Codificação Híbrida:** coordenadas dos semáforos são representadas por genes reais, enquanto os genes binários indicam o plano de sinalização para cada cruzamento.
- **Exemplo:** os genes [(-23.5505, -46.6333), 1, 0, 1, 1] podem representar a localização de um semáforo, onde o primeiro bit binário representa o plano de sinalização para o primeiro cruzamento (0 é vermelho, 1 é verde).

INICIALIZAÇÃO DA POPULAÇÃO

O método de inicialização desempenha um papel crucial em diversos algoritmos, influenciando diretamente a eficiência e a convergência do processo. Entre os métodos comuns de inicialização, destacam-se o aleatório e o hotstart.

Na inicialização aleatória, os parâmetros do modelo ou as soluções do algoritmo são atribuídos com valores de forma completamente randômica. Esse método visa introduzir diversidade na busca por soluções, explorando amplamente o espaço de busca. Apesar de sua simplicidade, a inicialização aleatória pode ser eficaz

em evitar a convergência prematura para ótimos locais e promover a exploração global.

Por outro lado, o hotstart, parte de uma estimativa inicial mais informada, muitas vezes derivada de resultados anteriores, conhecimento prévio do problema, ou métodos heurísticos determinísticos. Essa abordagem é especialmente útil quando se tem uma boa aproximação da solução, permitindo uma convergência mais rápida e eficiente. O hotstart capitaliza informações previamente obtidas, economizando recursos computacionais e acelerando o processo de otimização.

FUNÇÃO DE APTIDÃO (FITNESS)

A função de aptidão, muitas vezes referida como função fitness, desempenha um papel central em algoritmos genéticos e em problemas de otimização em geral. Essa função é fundamental para avaliar quão bem uma solução candidata resolve o problema em questão. Seu principal propósito é quantificar a qualidade relativa das soluções em uma população, orientando o processo de seleção e evolução.

A função de aptidão atribui um valor numérico a cada indivíduo da população, indicando o quão bem ele se adapta ao ambiente ou quão próximo está da solução ideal. Em problemas de maximização, a função de aptidão atribui valores mais altos a soluções melhores, enquanto em problemas de minimização, o oposto é verdadeiro. Essa função é, portanto, específica para cada problema e reflete os objetivos e critérios de desempenho desejados.

Ao longo das gerações, a seleção de indivíduos para reprodução é frequentemente guiada pela sua aptidão, com indivíduos mais aptos tendo uma probabilidade maior de serem escolhidos. Essa abordagem mimetiza o processo de seleção natural, onde organismos mais adaptados têm maior probabilidade de transmitir seus genes para as gerações seguintes. A formulação adequada da função de aptidão é crucial para o sucesso do algoritmo genético, influenciando diretamente a qualidade das soluções encontradas.

CRUZAMENTO

O cruzamento ou crossover, é uma operação fundamental em algoritmos genéticos, e diferentes métodos são empregados para combinar informações genéticas entre indivíduos. O método de crossover depende do tipo de codificação do problema, pois o resultado do cruzamento, precisa ser uma solução válida para o problema. Vamos ver alguns exemplos de crossover aplicados aos diferentes tipos de codificação.

CODIFICAÇÃO BINÁRIA: SINGLE-POINT CROSSOVER

- Um ponto de corte é escolhido aleatoriamente nos cromossomos dos pais.
- Os bits à esquerda do ponto de corte de um pai são combinados com os bits à direita do ponto de corte do outro pai para gerar descendentes.

Exemplo:

```
Pai 1:  11011010
Pai 2:  00100101

Ponto de Corte: 3

Filho 1: 110|00101
Filho 2: 001|11010
```

CODIFICAÇÃO REAL: ARITHMETIC CROSSOVER

- Para cada componente do vetor dos pais, o valor do filho é uma combinação linear dos valores dos pais, ponderada por uma constante alpha.

$$[\text{Filho}[i] = \alpha \times \text{Pai1}[i] + (1 - \alpha) \times \text{Pai2}[i]]$$

Para

- α = 0, filhos são cópias do Pai 2.
- α = 1, filhos são cópias do Pai 1.
- α = 0.5, filhos são média dos valores dos pais.

Exemplo:

```
Pai 1:  [1.5, 2.0, 3.0]
Pai 2:  [2.0, 1.8, 2.5]
Alpha: 0.7
     1.5*0.7 + 2*0.3 = 1.65
     1.5*0.3 + 2*0.7 = 1.85
Filho1: [1.65, 1.94, 2.85]
Filho2: [1.85, 1.86, 2.65]
```

Codificação Real: Uniform Crossover ou One-Point Crossover

- Para cada elemento nos vetores, uma decisão aleatória é tomada para manter o elemento do primeiro pai ou trocá-lo com o elemento do segundo pai.
- Esse processo é repetido para cada elemento independentemente, resultando em uma mistura de material genético de ambos os pais.

Exemplo:

```
P0 = [1, 2, 3]
P1 = [4, 5, 6]
Decisões aleatórias: [Manter, Trocar, Manter]
Filho 1 = [1, 5, 3]
Filho 2 = [4, 2, 6]
```

Codificação Combinatória: Order Crossover (OX1)

- Esse operador é projetado para preservar a ordem relativa dos elementos nas soluções, **garantindo que os descendentes gerados também sejam permutações válidas.**
- Em problemas combinatórios, não é permitido repetir os elementos na codificação.

- Método de cruzamento **não pode produzir soluções inválidas** para o problema.

Exemplo:

Dados os dois pais selecionados:

```
P0 = (A, B, C, D, E, F, G, H, I, J)
P1 = (B, D, A, H, J, C, E, G, F, I)
```

Selecione uma substring, por exemplo, entre 2 e 7 e inicialize os filhos com os genes de cada pai

```
F1 = (_, _, C, D, E, F, G, _, _, _)
F2 = (_, _, A, H, J, C, E, _, _, _)
```

Use os genes do outro pai para completar o gene dos filhos na ordem com que os genes aparecem no pai.

```
F1 = (B, A, C, D, E, F, G, H, J, I)
F2 = (B, D, A, H, J, C, E, F, G, I)
```

Os novos filhos preservam parcialmente a ordem dos pais e introduzem nova combinações.

Codificação Híbrida

- O crossover para codificação binária pode ser uma **mistura de diversos métodos de crossovers** que vimos para os diferentes tipos de dados da codificação do indivíduo,
- Cada tipo de dado dentro da codificação híbrida pode sofrer um tipo específico de cruzamento.
- A ideia central de todas as técnicas de crossover é criar novas soluções (filhos) por meio da combinação de soluções existentes (pais).
- O método de crossover precisa produzir **filhos que são soluções válidas para o problema**. Precisa respeitar as restrições do problema.

Os métodos mencionados acima são sugestões de métodos de crossover adequados aos diferentes tipos de codificação mencionados. No entanto, é importante ressaltar que não há uma regra fixa quanto à escolha desses métodos. A criatividade

na criação de abordagens distintas é encorajada, contudo, é crucial considerar dois aspectos fundamentais sobre o método:

- **Validade da Solução Gerada:** o método de crossover deve gerar soluções que sejam intrinsecamente válidas para o problema em questão. Por exemplo, no caso do “problema do caixeiro viajante”, a nova solução gerada deve visitar todas as cidades exatamente uma vez. Soluções que não percorrem todas as cidades ou revisitam uma cidade mais de uma vez são consideradas inválidas.
- **Custo Computacional:** a eficiência computacional do método de crossover é de extrema importância, dado que essa função será utilizada repetidamente em todas as gerações do algoritmo genético. Assim, a eficiência global do algoritmo depende, em grande medida, da eficiência desta função. Portanto, é desejável que o método de crossover seja computacionalmente eficiente para garantir um desempenho adequado ao longo de múltiplas iterações.

Ao considerar esses dois fatores, é possível desenvolver métodos de crossover inovadores que atendam às necessidades específicas de cada problema, promovendo a geração de soluções válidas e mantendo um desempenho eficiente do algoritmo genético como um todo.

MUTAÇÃO

A mutação é um operador genético fundamental na exploração pela busca de novas soluções. Ela atua como um mecanismo de introdução de variação no material genético, desempenhando um papel crucial na exploração do espaço de busca e evitando a convergência prematura para ótimos locais.

Dois parâmetros que controlam a mutação:

- **Probabilidade de mutação:** a probabilidade de um novo indivíduo, criado a partir do cruzamento dos pais, sofrer mutação.
- **Intensidade da mutação:** a intensidade com que a mutação alterará o valor dos genes do indivíduo. Uma mutação mais intensa significa alterar significativamente os valores da codificação do indivíduo, enquanto em uma

mutação menos intensa, o indivíduo sofrerá pequenas variações em seu conteúdo genético.

A escolha por ter maior ou menor intensidade e probabilidade de mutação depende da estratégia de convergência que se deseja adotar no algoritmo genético. Veja, a seguir, as relações de troca que temos ao optar por mais ou menos mutação.

- **Mais mutação:**
  - Mais exploração.
  - Menos aproveitamento.
  - Mais diversidade genética.
  - Risco de destruir boas soluções.
  - Menor chance de melhorar boas soluções já existentes.
- **Menos mutação:**
  - Menor exploração.
  - Maior aproveitamento.
  - Risco de convergência prematura. Maior chance de aperfeiçoar soluções que já são boas.

Vamos ver a seguir, algumas estratégias de mutação para as diferentes codificações que aprendemos:

Codificação Binária: Mutação Bit Flip

Seleciona aleatoriamente um ou mais bits e inverte seus valores.

```
Antes:  110101
Depois: 100101
```

Codificação Real: Mutação Gaussiana

Adiciona um valor aleatório com distribuição gaussiana ao gene. Quanto maior a intensidade da mutação, maior o intervalo da distribuição gaussiana, logo, maior

chance de uma mutação mais forte. Aleatoriamente, pode sair uma mutação muito forte com a probabilidade dada pela distribuição normal.

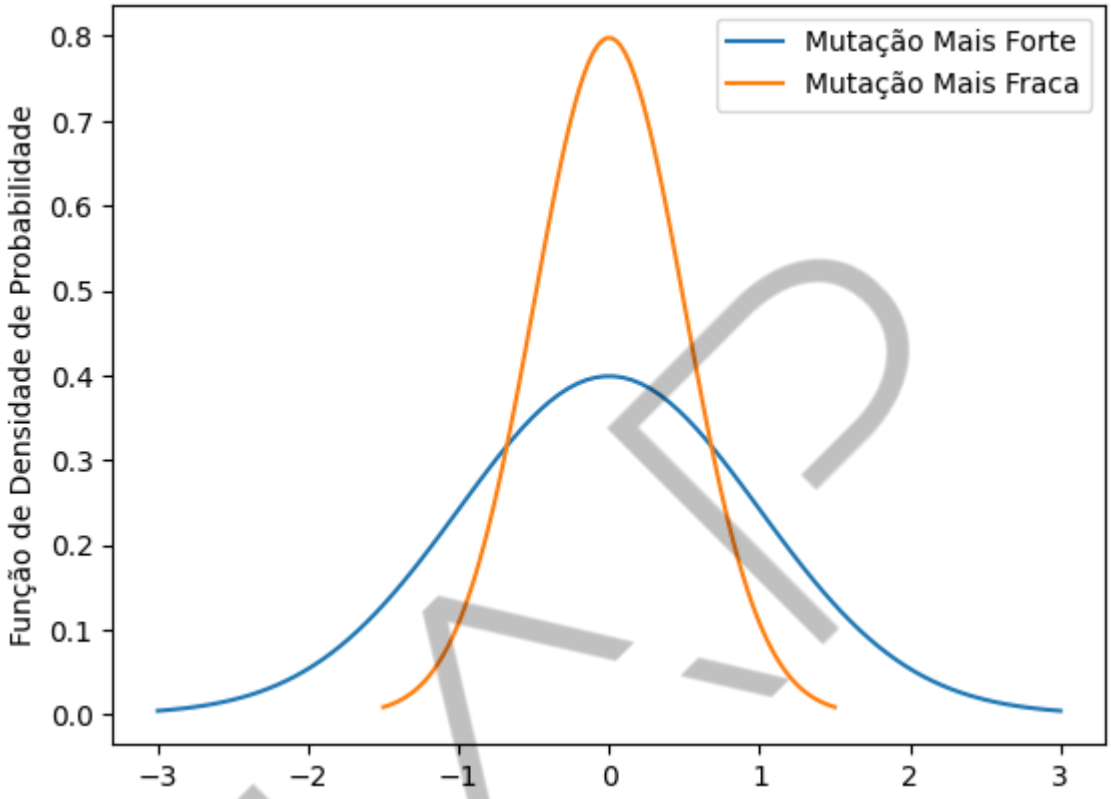

Figura 2 - Função de Densidade de Probabilidade Gaussiana
Fonte: elaborado pelo autor (2024), adaptado por FIAP (2024)

A curva azul representa uma mutação com maior intensidade, pois valores mais altos de mutação (eixo x) tem maior probabilidade de serem sorteados, quando comparados com a função de densidade de probabilidade laranja, onde valores mais baixos de mutação (eixo x) tem probabilidade muito maior de ocorrer. Compare o valor zero (não haver mutação), na curva laranja: ele possui 80% de chance de ser sorteado, enquanto que na curva azul, apenas 40%.

Exemplo:

```
Antes:  3.5

Mutação Fraca: 3.8

Mutação Forte: 5.6
```

Codificação Combinatória: Mutação por Inversão

Inverte a ordem de um subconjunto de genes no cromossomo. Quanto mais forte a mutação, maior o intervalo selecionado para ser invertido.

Note que essa mutação apenas inverte uma sequência de elementos, sendo uma mutação válida para o “problema do caixeiro viajante”, pois a solução não repete cidades e continua visitando todas.

Mutação forte:

- Para fazer uma mutação mais forte nesse método por inversão, basta selecionarmos trechos maiores para fazer a inversão.

```
Intervalo sorteado: [1, 5]
Antes:   [1, 2, 3, 4, 5, 6, 7, 8, 9]
Depois:  [1, 6, 5, 4, 3, 2, 7, 8, 9]
```

Mutação Fraca:

```
Intervalo sorteado: [1, 2]
Antes:   [1, 2, 3, 4, 5, 6, 7, 8, 9]
Depois:  [1, 3, 2, 4, 5, 6, 7, 8, 9]
```

Codificação Hibrida

Para a codificação hibrida, o método de mutação deve ser feito de maneira a atender o tipo de codificação do indivíduo. A codificação hibrida pode ser composta por diferentes tipos de dados, contendo uma parte real, outra combinatória, outra binária etc. Cada trecho da codificação hibrida representa um fenômeno da solução, com restrições personalizadas.

Assim, o método de mutação para a codificação híbrida será diversificado, utilizando várias funções, sendo cada uma adaptada para o tipo específico de codificação em cada parte do indivíduo.

Ao entendermos a função crucial de cada etapa, estaremos aptos e aptas a aplicar os conceitos aprendidos de forma eficaz em diferentes contextos práticos, e estaremos prontos e prontas para programar um algoritmo genético do zero na próxima aula!

CONCEITOS FUNDAMENTAIS EM ALGORITMOS GENÉTICOS

Vamos explorar alguns conceitos fundamentais dos algoritmos genéticos que são importantes para entender o funcionamento do algoritmo e como está sendo a sua performance.

- **Convergência:** refere-se à tendência do algoritmo genético de se aproximar de uma solução sub ótima ao longo das gerações.
- **Diversidade:** representa a variedade de soluções presentes na população, contribuindo para a exploração do espaço de busca.
- **Equilíbrio Convergência-Diversidade:** o desafio é encontrar um equilíbrio adequado entre convergência e diversidade. Muita convergência pode levar a soluções sub ótimas, enquanto muita diversidade pode retardar a convergência para a solução ótima.
- **Exploração:** busca de novas soluções em diferentes regiões do espaço de busca.
- **Aproveitamento:** foco em soluções promissoras para aprimorar a qualidade.

PARÂMETROS DOS ALGORITMOS GENÉTICOS

Tamanho da População

O tamanho da população em algoritmos genéticos refere-se à quantidade de indivíduos presentes em cada geração durante a execução do algoritmo. É um parâmetro crítico que influencia diretamente a dinâmica da busca e otimização realizada pelo algoritmo genético.

Valor Ideal

Não há um valor ideal universal para o tamanho da população, pois isso depende da natureza específica do problema que está sendo abordado. Em geral, um

tamanho de população maior favorece uma exploração mais abrangente do espaço de busca, permitindo que o algoritmo descubra uma variedade maior de soluções potenciais. No entanto, um tamanho de população muito grande pode levar a um aumento significativo nos custos computacionais. O aumento do custo computacional entre cada geração diminui a velocidade da evolução das gerações.

Impacto do Tamanho da População

População Grande

Vantagens:

- Maior diversidade genética.
- Exploração eficiente de diferentes regiões do espaço de busca.

Desvantagens:

- Pode levar a uma convergência mais lenta, pois, a cada geração, há mais indivíduos para avaliar.
- Pode resultar em um alto consumo de recursos computacionais sem melhorias significativas no espaço de busca.

População Pequena:

Vantagens:

- Menor uso de recursos computacionais.
- Convergência potencialmente mais rápida, pois processa rapidamente mais gerações.

Desvantagens:

- Menor diversidade genética. Pode ficar “presa” em ótimos locais, não explorando mais amplamente o espaço de soluções.
- Possibilidade de convergência prematura para uma solução sub ótima.

Ajuste Dinâmico

Uma abordagem comum é realizar ajustes dinâmicos no tamanho da população com base no desempenho do algoritmo ao longo do tempo. Isso pode envolver a redução da população após a convergência ter começado ou a expansão se a diversidade genética estiver diminuindo. Alguns exemplos de ajustes dinâmicos:

Redução Gradual do Tamanho da População: quando sinais de convergência são detectados, considera-se a redução gradual do tamanho da população. Isso visa concentrar recursos computacionais nas melhores soluções encontradas, acelerando o processo de convergência.

Expansão em Caso de Estagnação: se a diversidade genética diminuir ou o desempenho estagnar, pode-se considerar a expansão da população. Isso reintroduz diversidade, explorando novas regiões do espaço de busca e evitando convergência prematura.

Taxa de Crossover

A taxa de crossover determina a probabilidade de dois indivíduos trocarem material genético durante a reprodução. Uma taxa alta pode acelerar a convergência, enquanto uma taxa baixa pode preservar a diversidade.

Equilíbrio entre Exploração e Explotação: a taxa de crossover influencia o equilíbrio entre explorar novas soluções e aproveitar as soluções boas já existentes. Taxas de crossover baixas, acontecem quando um filho é uma cópia do pai, ou muito semelhante, com pouca característica herdada do segundo pai. Nesse caso, as soluções são mais preservadas (aproveitamento) e acontece menor exploração de soluções novas (exploração).

Taxas de crossover altas acontecem quando os filhos são pouco parecidos com apenas um dos pais, ou seja, é uma mistura equilibrada entre os dois pais. Nesse caso, são favorecidas as novas soluções (exploração) em detrimento da manutenção das soluções boas já encontradas (aproveitamento).

A taxa de crossover pode representar o quanto os filhos serão parecidos com os pais, por exemplo:0.9 significa que o filho1 será 90% igual ao pai1 e 10% igual ao pai2, e o filho2 será 10% parecido com o pai1 e 90% com o pai2. Ou pode significar a probabilidade de ocorrer o cruzamento. Nesse caso, para 0.5 significa que há uma chance de 50% dos filhos serem feitos a partir do cruzamento dos pais (cruzamento) e 50% de chance de serem uma cópia exata dos pais (não cruzamento).

Adaptação Dinâmica: a adaptação dinâmica da taxa de crossover em algoritmos genéticos ocorre com ajustes estratégicos para mais ou para menos com base em condições específicas do problema. Quando o algoritmo apresenta uma convergência lenta ou a diversidade genética diminui, um aumento na taxa de

crossover é considerado para promover uma exploração mais abrangente do espaço de busca. Esse ajuste estimula a introdução de novas combinações genéticas e acelera a convergência. Por outro lado, se a convergência é rápida demais ou há uma preocupação com a estagnação em soluções sub ótimas, uma redução na taxa de crossover pode preservar características valiosas na população, permitindo uma exploração mais cuidadosa antes da convergência final. A estratégia dinâmica busca equilibrar eficazmente a convergência e a diversidade, adaptando-se continuamente ao desempenho e às características evolutivas do algoritmo.

Taxa de Mutação

A taxa de mutação representa a probabilidade de um gene sofrer uma alteração aleatória. Quanto mais mutação, mais soluções novas aparecerão, e maior será a exploração de novas soluções (exploração). Porém, menor será o aproveitamento das boas soluções, pois elas podem ser perdidas ao sofrer uma nova mutação (aproveitamento).

Ajuste dinâmico: visa otimizar a exploração do espaço de busca genético ao longo das gerações. Este ajuste ocorre em resposta ao desempenho do algoritmo, adaptando-se às características específicas do problema. Quando a convergência é lenta ou a diversidade genética é insuficiente, um aumento na taxa de mutação é considerado. Isso estimula a introdução de variações genéticas, promovendo uma exploração mais abrangente do espaço de soluções. Em contrapartida, se a convergência é rápida demais ou há risco de convergência prematura para soluções sub ótimas, uma redução na taxa de mutação é aplicada. Isso ajuda a preservar características valiosas e a evitar modificações excessivas que poderiam comprometer a qualidade da solução. A estratégia dinâmica da taxa de mutação busca alcançar um equilíbrio ótimo entre a exploração e a explotação, adaptando-se de forma inteligente às necessidades evolutivas do algoritmo genético durante o processo de otimização.

Métricas de desempenho de Algoritmos Genéticos

Convergência

Acompanhar a convergência do algoritmo ao longo das gerações. Métricas como a média de aptidão e a melhor aptidão na população podem indicar quão rapidamente o algoritmo está se aproximando de uma solução ótima.

Média de Aptidão (Fitness Mean):

Calcule a média das aptidões da população a cada geração. Uma convergência eficaz geralmente resulta em uma diminuição consistente da média de aptidão em direção a valores ótimos.

Melhor Aptidão (Best Fitness):

Acompanhe a evolução da melhor aptidão na população. A convergência é evidenciada por melhorias contínuas nessa métrica, indicando a aproximação a uma solução ótima.

Desvio Padrão da Aptidão (Fitness Standard Deviation):

Calcule o desvio padrão das aptidões na população. Uma redução no desvio padrão ao longo das gerações sugere convergência, indicando que a população está se aproximando de soluções semelhantes.

Diversidade

Avaliar a diversidade genética presente na população. A falta de diversidade pode levar à convergência prematura. Métricas de diversidade, como a distância euclidiana entre soluções, oferecem insights sobre a exploração do espaço de busca.

Distância Euclidiana Entre Soluções:

Meça a distância euclidiana média entre pares de soluções na população. Um aumento na distância indica maior diversidade genética.

Entropia Genética:

Utilize a entropia genética para quantificar a diversidade na população. Uma entropia mais alta sugere uma população mais diversificada.

A entropia genética é uma medida que quantifica a diversidade genética dentro de uma população em algoritmos genéticos. Ela é derivada do conceito de entropia

da teoria da informação e é utilizada para avaliar a distribuição de alelos (variantes genéticas) na população. No contexto de algoritmos genéticos, a entropia genética oferece insights sobre quão variada é a informação genética presente na população.

A fórmula básica para calcular a entropia genética é análoga à fórmula de entropia na teoria da informação e pode ser expressa da seguinte forma:

$$[H = -\sum_{i=1}^{n} p_i \cdot \log_2(p_i)]$$

Onde:

- H é a entropia genética,
- n é o número de alelos distintos na população, e
- p_i é a proporção do alelo i na população.

Para o “problema do Caixeiro Viajante (TSP)”, por exemplo, os alelos representam diferentes rotas ou ordens de visitação das cidades. Uma entropia genética mais alta indica uma população mais diversificada, com diferentes soluções representadas. Por outro lado, uma entropia mais baixa sugere uma convergência para um conjunto mais restrito de soluções.

Vamos considerar um exemplo simplificado do “problema do Caixeiro Viajante” (TSP) com um conjunto de cidades e rotas possíveis. Suponhamos que temos 4 cidades (A, B, C, D) e uma população de soluções representadas por diferentes ordens de visita dessas cidades.

Vamos representar a população inicial por três soluções (rotas):

Rota 1: A -> B -> C -> D

Rota 2: C -> D -> A -> B

Rota 3: B -> A -> D -> C

Para calcular a entropia genética, primeiro precisamos contar a frequência de cada alelo (ordem de visita das cidades) na população. Neste caso, os alelos são as

diferentes ordens de visita das cidades. A entropia genética é então calculada usando a fórmula mencionada anteriormente.

Suponha que, após a contagem, encontramos as seguintes frequências (proporções):

Frequência de Rota 1: $p_1 = \frac{1}{3}$

Frequência de Rota 2: $p_2 = \frac{1}{3}$

Frequência de Rota 3: $p_3 = \frac{1}{3}$

Substituindo esses valores na fórmula de entropia genética:

$$[H = -\left(\frac{1}{3} \cdot \log_2\left(\frac{1}{3}\right) + \frac{1}{3} \cdot \log_2\left(\frac{1}{3}\right) + \frac{1}{3} \cdot \log_2\left(\frac{1}{3}\right)\right)]$$

$$[H \approx 1.585]$$

Neste exemplo, a entropia genética é aproximadamente 1.585, indicando uma população relativamente diversificada em termos de ordens de visita das cidades.

Quanto **maior o valor da entropia, maior a diversidade genética** na população. Este é um exemplo simplificado, e em problemas mais complexos, o cálculo da entropia genética envolveria mais soluções e alelos.

Número de Soluções Únicas:

Contabilize o número de soluções únicas na população. Um declínio nesse número pode indicar uma convergência para um grupo restrito de soluções.

Análise de Frentes de Pareto (para Problemas Multi-Objetivo)

Avalie a convergência e a diversidade considerando as frentes de Pareto. Uma frente de Pareto bem distribuída indica uma boa cobertura do conjunto de soluções não dominadas.

Eficiência Computacional

Medir o desempenho computacional do algoritmo, como o tempo de execução de cada geração, e o tempo de execução específico de cada operador genético como cálculo do fitness, crossover, mutação etc. Otimizar a eficiência é crucial, especialmente para problemas computacionalmente intensivos. Quanto mais rápido cada geração rodar, mais rápido o algoritmo convergir para uma solução otimizada.

Alguns passos do cálculo entre cada geração podem ser paralelizados, como o cruzamento e a mutação. Considere usar processamento paralelo para problemas computacionalmente intenso.

O QUE VOCÊ VIU NESTA AULA?

Na terceira aula, exploramos os princípios e conceitos fundamentais dos Algoritmos Genéticos (AGs). Iniciamos com uma apresentação detalhada do funcionamento do algoritmo genético, revisitando os princípios discutidos na aula anterior. Introduzimos um diagrama abrangente que guiou cada etapa do AG, desde o início até o fim da aula, proporcionando uma visão clara e sequencial do processo evolutivo. Exploramos minuciosamente cada etapa, começando pela representação de soluções por meio de codificação genética, seguida pela inicialização da população e a crucial avaliação da população por meio da função de aptidão (“Fitness Function”). Investigamos a aplicação de operadores genéticos, como crossover e mutação, destacando sua importância na introdução de variabilidade nas soluções, um elemento-chave para a eficácia do AG.

Agora, gostaríamos de saber a sua opinião sobre o conteúdo! Compartilhe seus comentários e dúvidas no Discord, onde estamos disponíveis na comunidade para responder a perguntas, promover networking, fornecer avisos e muito mais. Junte-se a nós!

REFERÊNCIAS

CHENEY, N.; MACCURDY, R.; CLUNE, J.; LIPSON, H. **Unshackling Evolution: Envolving Soft Robots with Multiple Materials and a Powerful Generative Encoding**. Cornell University. [s.d]. Disponível em: http://jeffclune.com/publications/2013_Softbots_GECCO.pdf. Acesso em: 25 mar. 2024.

POLIMANTE, S.; PRATI, R.; KLEINSCHMIDT, J.H. **Evolução multiobjetivo de trajetórias como múltiplas curvas de Bézier para VANTs**. 2020. [s.l.]. Disponível em: https://www.researchgate.net/publication/376134695_Evolucao_multiobjetivo_de_trajetorias_como_multiplas_curvas_de_Bezier_para_VANTs. Acesso em: 25 mar. 2024.

POLIMANTE, S.; PRATI, R.; KLEINSCHMIDT, J.H. **Otimização Multiobjetivo de Trajetórias de VANTs Utilizando Curvas de Bézier e Algoritmos Genéticos**. Conference XIV Brazilian Congress of Computational Intelligence, Belém, 2019. Disponível: em: https://www.researchgate.net/publication/337051141_Otimizacao_Multiobjetivo_de_Trajetorias_de_VANTs_Utilizando_Curvas_de_Bezier_e_Algoritmos_Geneticos. Acesso em: 25 mar. 2024.

STANLEY, O. K.; MIIKULAINEN, R.; et.al. **Envolving Neural Networks through Augmenting Topologies.** IEEE Explore – MIT Press. 2002. Disponível em: https://ieeexplore.ieee.org/document/6790655. Acesso em: 25 mar. 2024.

SUN, Y.; XUE, B.; ZHANG, M. YEN, G.G. **Envolving Deep Convolutional Neural Networks for Image Classification**. Cornell University. 2017. Disponível em: https://arxiv.org/abs/1710.10741. Acesso em: 25 mar. 2024.

TANGHE, K. B. **On the origin of species: the story of Darwin's title**. The Royal Society Journal of the History of Science. 2018. Disponível em: https://royalsocietypublishing.org/doi/10.1098/rsnr.2018.0015. Acesso em: 25 mar. 2024.

PALAVRAS-CHAVE

Genetic Algorithms. Neural Networks. Neuroevolution.