UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE FÍSICA DE SÃO CARLOS

Fabio Aparecido Cavalheiro

Algoritmos genéticos como ferramenta de otimização das interconexões da rede de Hopfield atuando como memória associativa

São Carlos

2021

Fabio Aparecido Cavalheiro

Algoritmos genéticos como ferramenta de otimização das interconexões da rede de Hopfield atuando como memória associativa

Trabalho de conclusão de curso apresentado ao Programa de Graduação em Física do Instituto de Física de São Carlos da Universidade de São Paulo, para obtenção do título de Bacharel em Física Computacional.

Orientador: Prof. Dr. Luciano da Fontoura Costa

Coorientador: Prof. Dr. Gonzalo Travieso

São Carlos

2021

Cavalheiro, Fabio Aparecido
Algoritmos genéticos como ferramenta de otimização das interconexões da rede de Hopfield atuando como memória associativa / Fabio Aparecido Cavalheiro; orientador Luciano da Fontoura Costa; co-orientador Gonzalo Travieso -- São Carlos, 2021.
27 p.

Trabalho de Conclusão de Curso (Bacharel em Física Computacional) -- Instituto de Física de São Carlos, Universidade de São Paulo, 2021.

1. Rede neural de Hopfield. 2. Algoritmo genético. 3. Otimização. I. Costa, Luciano da Fontoura, orient. II. Travieso, Gonzalo, co-orient. III. Título.

RESUMO

CAVALHEIRO, F. A. **Algoritmos genéticos como ferramenta de otimização das interconexões da rede de Hopfield atuando como memória associativa**. 2021. 27p. Monografia (Trabalho de Conclusão de Curso) - Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, 2021.

Uma das características mais distintivas da rede neural de Hopfield está relacionada ao conceito de memória associativa, que é a capacidade da rede neural em recuperar um determinado padrão previamente armazenado a partir de sua versão perturbada ou incompleta. O problema é que dependendo do tipo de padrão utilizado, como por exemplo padrões correlacionados, a rede de Hopfield sofre uma séria deficiência em seu desempenho. Portanto este trabalho propõe a implementação de um algoritmo genético paralelo como ferramenta para otimizar as interconexões da rede de Hopfield previamente treinada com dados do mundo real, caracteres manuscritos extraídos da base de dados *MNIST*. A implementação da rede neural e do algoritmo genético foram feitas utilizando a linguagem de programação *C++* e a paralelização foi feita através da interface de programação conhecida como *OpenMP*. Foi confirmada a eficácia da utilização do algoritmo genético em auxiliar a rede de Hopfield atuando como memória associativa, onde foi obtido como pior resultado uma taxa de acerto de 99% no número de pixels corretamente recuperados a partir da versão incompleta de um dos padrões previamente armazenados na rede. Também foi realizado um estudo demonstrando que os parâmetros de inicialização do algoritmo genético tem grande influência nos resultados obtidos.

Palavras-chave: Rede neural de Hopfield. Algoritmo genético. Otimização.

SUMÁRIO

1 INTRODUÇÃO

Redes neurais são algoritmos criados como uma forma de tentar emular as operações do cérebro humano, permitindo que programas de computadores reconheçam padrões e resolvam problemas nas áreas de inteligência artificial. Como exemplo de aplicação, podemos citar os benefícios do uso da inteligência artificial na medicina, permitindo uma maior agilidade no diagnóstico de doenças através do uso de redes neurais processando as imagens coletadas em exames médicos.

Existem vários modelos de redes neurais, estes diferem-se quanto à arquitetura e ao processo de aprendizado e cada modelo é mais adequado para resolver um determinado tipo de problema. Dentre essas redes neurais temos a rede de Hopfield, que trabalha com o conceito de memória associativa. Memórias associativas são modelos inspirados na habilidade do cérebro humano de recordar por associação, sendo assim possível recuperar uma memória a partir de sua versão incompleta ou perturbada.

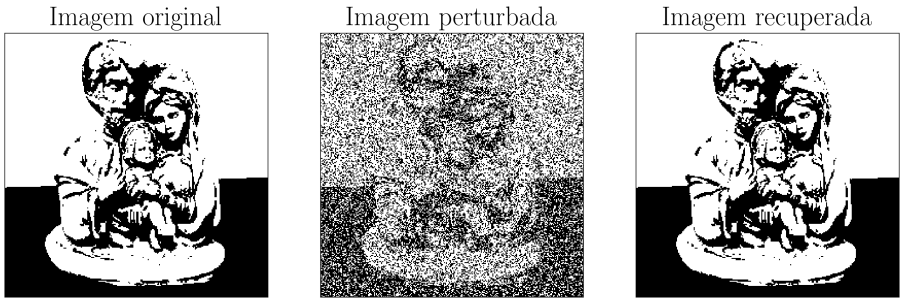

Figura 1 – Exemplo da rede de Hopfield atuando como memória associativa.
Fonte: Elaborada pelo autor.

Mas a rede de Hopfield apresenta diversas limitações, onde a qualidade dos resultados dependem por exemplo, da semelhança das imagens armazenadas, da quantidade de imagens armazenadas, do tamanho da rede. Existem diversos estudos sobre como é possível sanar essas deficiências (1), e neste trabalho será proposto uma solução que fará o uso do algoritmo genético paralelo como forma de otimizar as interconexões da rede de Hopfield atuando como memória associativa.

O trabalho está organizado da seguinte forma, no capítulo 2 é feita uma breve revisão teórica dos conceitos envolvendo redes neurais e o algoritmo genético, no capítulo 3 é mostrado em detalhes a metodologia usada, no capítulo 4 é apresentado os resultados obtidos junto com a discussão e por fim no capítulo 5 temos a conclusão do trabalho e sugestões de trabalhos futuros.

2 REFERENCIAL TEÓRICO

2.1 Modelo de McCulloch-Pitts

Os neurônios são as unidades básicas do sistema nervoso e geram os sinais elétricos que transmitem rapidamente informações por longas distâncias.

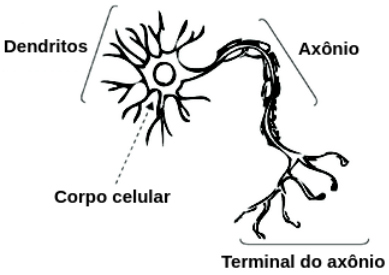

Figura 2 – Representação simplificada do neurônio biológico.
Fonte: Adaptada de MENG. (2)

Os dendritos têm como função receber os sinais elétricos oriundos de outros neurônios e conduzi-los até o corpo celular. Ali, a informação é processada e novos sinais elétricos são gerados. Estes sinais são transmitidos a outros neurônios, passando pelo axônio e atingindo o terminal do axônio, onde é estabelecida a conexão sináptica com os neurônios seguintes.

O primeiro modelo computacional de um neurônio foi proposto pelos neurocientistas Warren McCulloch e Walter Pitts.

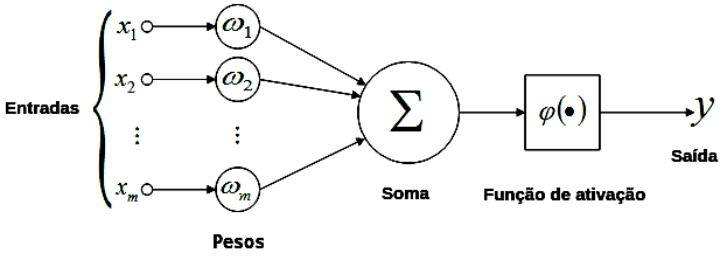

Figura 3 – Modelo do neurônio artificial proposto por McCulloch-Pitts.
Fonte: Adaptada de OLIVEIRA. (3)

O modelo de McCulloch-Pitts realiza uma soma ponderada dos valores de entrada dos outros neurônios e retorna um valor de saída, que dependendo se a soma está acima ou abaixo de um limiar de disparo, será 1 ou 0 respectivamente. A escolha é feita pela função de ativação e os valores 1 e 0 representam os estados “disparando” e “não disparando”.

2.2 Teoria Hebbiana

Em 1949, Donald Hebb propôs em seu livro, *The Organization of Behavior*, o que veio a ser conhecido como a regra de Hebb (4):

"Quando um axônio de uma célula A está próximo o suficiente de excitar a célula B e repetidamente ou persistentemente participa da ativação desta, um processo de crescimento ou mudança metabólica ocorre em uma ou ambas as células, de tal forma que a eficiência de A ativar B é aumentada."

Portanto Hebb propôs não apenas que, quando dois neurônios disparam juntos, a conexão entre os neurônios é fortalecida, mas também que essa atividade é uma das operações fundamentais necessárias para o aprendizado e a memória.

2.3 Modelo de Hopfield

Em 1982, John Hopfield publicou um artigo (5), na qual ele faz um estudo do comportamento de redes recorrentes com conexões sinápticas através da definição de uma função de energia.

No modelo de Hopfield temos N neurônios totalmente conectados entre si, estabelecendo N^2 conexões sinápticas que podem ser entendidas como os elementos de memória da rede, onde a informação previamente aprendida ou armazenada é mantida. Os pesos são alterados conforme a rede aprende novos padrões. (6)

O processo de aprendizagem da rede é definido pela regra de Hebb:

$$w_{ij} = \frac{1}{N}\sum_{\mu=1}^{p} \xi_i^\mu \xi_j^\mu \quad (i \neq j), w_{ii} = 0 \tag{2.1}$$

Onde p representa o número de padrões a serem armazenados e ξ_i^μ com $i = 1, 2, ..., N$ e $\mu = 1, 2, ..., p$ representam o i-nésimo estado do μ-nésimo padrão e é 1 ou -1.

Quando uma versão perturbada ou incompleta de um padrão previamente armazenado é fornecida para a rede, os estados dos neurônios passam por uma atualização. Essa atualização pode ser feita de duas formas: sincronizada, atualizando o estado de todos os neurônios simultaneamente a cada intervalo de tempo ou não sincronizada, atualizando o estado de um neurônio por intervalo de tempo. Neste trabalho foi considerado o caso não sincronizado utilizando a seguinte equação:

$$S_i^\mu(t+1) = shl\left(\sum_{j\neq i}^{N} w_{ij} S_j^\mu(t)\right) \tag{2.2}$$

Onde $S_i^{\mu}(t)$ é o estado do i-nésimo neurônio no tempo t e $shl(x)$ significa *symmetric hard limit*, cuja definição é:

$$shl(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases} \tag{2.3}$$

Durante o processo de atualização, quando os estados dos neurônios da rede ficam constantes ou quase constantes, foi atingido um ponto fixo na dinâmica da rede e assim obtemos o padrão recuperado.

Uma das contribuições mais importantes do artigo que Hopfield publicou em 1982 foi a definição da função de energia:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} S_i S_j \tag{2.4}$$

A característica mais importante da função de energia é que ela sempre diminui, ou permanece quase constante, quando o sistema evolui de acordo com sua dinâmica. Assim a rede será atualizada até convergir para um mínimo local de energia. Esses pontos mínimos são chamados de atratores ou pontos fixos do sistema e definem os padrões armazenados pela rede.

2.4 Algoritmo genético

O algoritmo genético é um método de otimização de inspiração biológica que teve como um de seus principais estudiosos John Henry Holland. Segundo a teoria evolucionária de Charles Darwin, os seres mais aptos possuem melhores chances de sobrevivência, passando assim seus genes para seus descendentes de tal forma os indivíduos das gerações futuras possuirão uma aptidão melhor que seus antepassados.

Na implementação do algoritmo genético é criada uma população inicial de indivíduos, aleatoriamente ou a partir de uma solução sub-otimizada, onde cada indivíduo representa uma solução para um determinado problema, assim indivíduos que apresentam uma performance melhor na solução do problema são selecionados para gerar a próxima geração através da implementação de operadores genéticos como crossover e mutação.

Este processo de escolha dos melhores indivíduos e aplicação dos operadores genéticos é repetido por um determinado número de gerações, até que no fim é extraído o melhor indivíduo da população final, indivíduo este que é a solução otimizada para o problema inicial proposto.

3 METODOLOGIA

3.1 Base de dados MNIST

Neste trabalho foi utilizado a base de dados MNIST, que é composta por 70 mil dígitos manuscritos (0 a 9), divididos em 60 mil exemplos para treinamento e 10 mil exemplos para testes. É uma base de dados amplamente utilizada em diversos métodos de reconhecimento de padrões em dados do mundo real. Os dígitos foram normalizados por tamanho e centralizados em uma imagem de tamanho fixo, possuindo uma dimensão de 28x28 pixels.

Como os dígitos da base de dados são apresentados em tons de cinza, os valores dos pixels variam de 0 a 255. Para a utilização desses dados no trabalho foi necessário fazer uma conversão para que os valores dos pixels sejam bipolares -1 e 1. Os dígitos também foram reduzidos para o tamanho de 14x14 pixels.

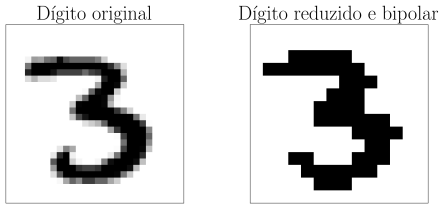

Figura 4 – Processo de transformação dos dígitos da base MNIST.
Fonte: Elaborada pelo autor.

Os 10 dígitos escolhidos para serem armazenados na rede de Hopfield foram:

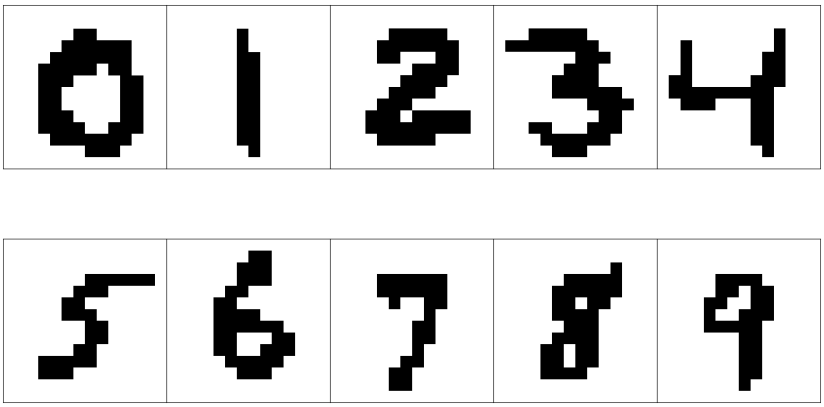

Figura 5 – Conjunto de dígitos usados no treinamento da rede.
Fonte: Elaborada pelo autor.

3.2 Treinamento da rede

Como cada padrão possui um tamanho de 14x14 pixels, a rede implementada terá um tamanho de 196 neurônios mutualmente conectados.

Para o processo de armazenamento desses padrões, cada um deles foram transformados em um vetor enfileirando as linhas das matrizes de cada dígito. Estes vetores são reunidos em um arquivo de entrada, onde cada linha representa um padrão a ser armazenado na rede.

Este arquivo é lido pelo programa desenvolvido em *C++* que implementando a regra de Hebb através da equação 2.1, gera a matriz de pesos sinápticos. Por fim o padrão incompleto ou perturbado de um dos padrões previamente armazenados na rede é lido de um segundo arquivo.

3.3 Geração da população inicial

Na implementação do algoritmo genético não é gerada uma população inicial com indivíduos aleatórios, como geralmente é feito em outras aplicações do algoritmo genético, aqui partiremos de um indivíduo sub-otimizado, que é a matriz de pesos sinápticos gerada durante o treinamento da rede de Hopfield.

Assim cada novo indivíduo da população inicial é gerado multiplicando cada elemento do indivíduo sub-otimizado por um número x, onde $x \in \{-1, 0, 1\}$. Quando é escolhido $x = 0$ é podado a conexão entre os neurônios, quando $x = -1$ é invertida a relação de excitação/inibição dos neurônios e no último caso $x = 1$ é conservada a conexão sináptica.

3.4 Definição do fitness

Com a população inicial criada, devemos classificar cada um dos indivíduos de acordo com um mérito ou fitness. O fitness escolhido foi a eficiência no processo de recuperar um padrão $\boldsymbol{v}_0$ como $\boldsymbol{v}_{rec}$ a partir de sua versão perturbada $\boldsymbol{v}_{pet}$. (7)

$$F(\boldsymbol{v}_{rec}, \boldsymbol{v}_0) = \frac{1}{N}\langle \boldsymbol{v}_{rec}, \boldsymbol{v}_0 \rangle \tag{3.1}$$

Onde N é o número de neurônios da rede e $-1 \leq F(\boldsymbol{v}_{rec}, \boldsymbol{v}_0) \leq 1$, assim caso todos os elementos serem corretamente recuperados, temos $F(\boldsymbol{v}_{rec}, \boldsymbol{v}_0) = 1$ e no caso de todos estarem errados, $F(\boldsymbol{v}_{rec}, \boldsymbol{v}_0) = -1$.

Como a expressão acima pode ser vista como a semelhança entre dois vetores, ela também foi utilizada para identificar a qual padrão originalmente armazenado na rede refere-se a versão perturbada que é lida do arquivo de entrada. Assim, antes da aplicação do

algoritmo genético, o valor de F é calculado para todos os padrões que foram armazenados na rede e aquele que apresentar o maior valor é considerado como a versão original $\boldsymbol{v}_0$.

3.5 Processo de escolha dos melhores indivíduos

Os indivíduos, que são as matrizes de pesos sinápticos, que apresentam os maiores valores de fitness são selecionados para gerar a próxima geração. Isto é feito escolhendo aleatoriamente dois pais do conjunto dos 40% melhores indivíduos da população e a partir do operador genético de crossover é gerado dois filhos que irão substituir o restante dos 60% da população. O tamanho da população é mantido constante durante todo o processo de evolução.

3.6 Mutação e crossover

O operador de crossover é implementado nos elementos das matrizes de pesos sinápticos dos indivíduos que foram selecionados de acordo com o método descrito anteriormente. A escolha de qual elemento vai para cada filho é feita de forma aleatória, mas sempre respeitando que cada elemento de um dos pais vai para somente um dos filhos.

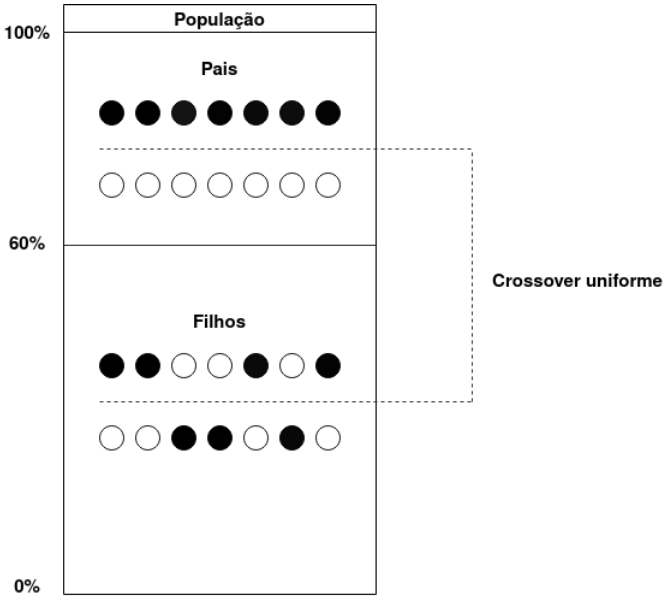

Figura 6 – Detalhes da implementação do operador genético crossover. A população é ordenada de forma decrescente em relação ao valor do fitness.
Fonte: Elaborada pelo autor.

O operador genético de mutação é aplicado após a etapa do crossover e é definido pelo parâmetro *taxa de mutação*. Durante o processo de mutação é gerado um número aleatório entre 0 e 1 para cada um dos filhos criados e caso esse número seja menor que a taxa de mutação, o indivíduo em questão sofre a operação de mutação, onde dois elementos escolhidos de forma arbitrária trocam de posição.

3.7 Paralelização do algoritmo genético

A ferramenta utilizada na paralelização foi o *OpenMP*, que é um conjunto de anotações que é feito no código sequencial de um programa para auxiliar o compilador em identificar quais regiões podem ser paralelizadas. A partir daí o próprio compilador se encarrega do processo de paralelização, criando e controlando a execução das threads. O *OpenMP* trabalha com o conceito de *fork and join*.

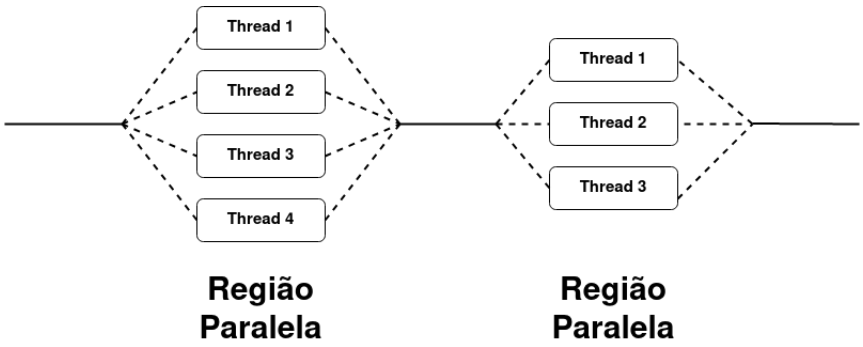

Figura 7 – Modelo de fork and join.
Fonte: Elaborada pelo autor.

Quando o programa passa de uma região sequencial para uma região paralela, temos um *fork*, nessa região existem diversas threads executando simultaneamente com um alto grau de independência uma da outra, quando as threads terminam a sua execução temos o *join*, situação na qual se passa de uma região paralela para uma região sequencial. As anotações mencionadas anteriormente é justamente para mostrar ao compilador quais partes devem ser executadas em paralelo.

Portanto o algoritmo genético paralelo consiste na distribuição das tarefas de um algoritmo genético simples por diferentes threads, que são executadas em processadores distintos, sendo que a ideia básica é que cada thread possa criar novos indivíduos através da aplicação dos operadores genéticos e calcular os fitness dos indivíduos da população em paralelo.

4 RESULTADOS

4.1 Sem o algoritmo genético

O conjunto de 10 padrões mostrados na seção 3.1 foram armazenados na rede e os padrões perturbados correspondem as respectivas versões incompletas de cada dígito. Utilizando a equação 3.1 para determinar o fitness de cada dígito, foi criada a tabela 1 mostrando os resultados obtidos, lembrando que o fitness igual a 1 corresponde ao padrão perfeitamente recuperado.

Tabela 1 – Resultados obtidos sem o algoritmo genético.

Dígito	Fitness	Pixels errados	Taxa de acerto
0	0.448980	54	72.4%
1	0.775510	22	88.8%
2	0.632653	36	81.6%
3	0.591837	40	79.6%
4	0.551020	44	77.6%
5	0.765306	23	88.3%
6	0.673469	32	83.7%
7	0.877551	12	93.9%
8	0.857143	14	92.9%
9	0.867347	13	93.4%

Fonte: Elaborada pelo autor.

Embora os resultados obtidos apresentam uma alta taxa de acertos, olhando para o caso específico do dígito 4, temos que o padrão recuperado não se assemelha em nada ao original.

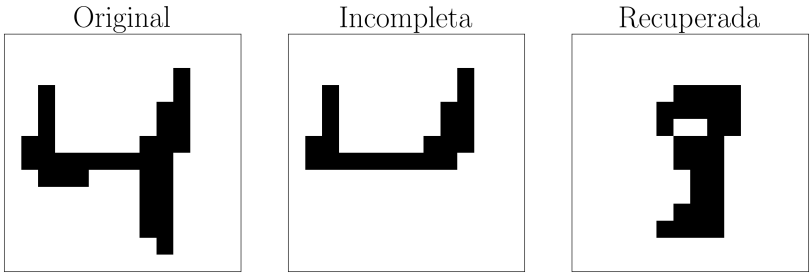

Figura 8 – Padrão não recuperado corretamente.
Fonte: Elaborada pelo autor.

Isto ocorre devido aos caracteres manuscritos serem correlacionados, criando assim estados espúrios. Estados espúrios são mínimos locais da função de energia que não correspondem a nenhum dos padrões armazenados.

4.2 Com o algoritmo genético

Durante a aplicação do algoritmo genético, os parâmetros foram mantidos constantes para termos uma base de comparação no desempenho do algoritmo para cada dígito. Os parâmetros usados foram:

- Tamanho da população: 400
- Número de gerações: 200
- Taxa de mutação: 0.2
- Aleatoriedade na criação da população inicial (β): 0.65

Dos parâmetros acima, *aleatoriedade na criação da população inicial* (β), representa o quão semelhante os indivíduos que são criados para a população inicial são do indivíduo original, que no caso é a matriz de pesos ou conexões sinápticas. Este parâmetro foi de extrema importância na obtenção dos resultados desse trabalho, portanto a escolha deste parâmetro é discutida com mais detalhes na seção 4.3. A tabela 2 apresenta os resultados obtidos.

Tabela 2 – Resultados obtidos usando o algoritmo genético.

Dígito	Fitness	Pixels errados	Taxa de acerto
0	0.989796	1	99.5%
1	0.979592	2	99%
2	1	0	100%
3	1	0	100%
4	0.989796	1	99.5%
5	1	0	100%
6	1	0	100%
7	0.989796	1	99.5%
8	1	0	100%
9	1	0	100%

Fonte: Elaborada pelo autor.

No pior caso foi obtido uma taxa de acerto de 99% no número de pixels corretamente recuperados para o dígito 1, que em comparação ao resultado encontrado sem o uso do algoritmo genético, a taxa de acerto foi de 88.8%, portanto os resultados obtidos comprovam a eficácia do algoritmo genético em auxiliar a rede de Hopfield. Na figura abaixo a primeira coluna representa os padrões armazenados na rede, a segunda coluna as versões incompletas e a terceira coluna os padrões recuperados.

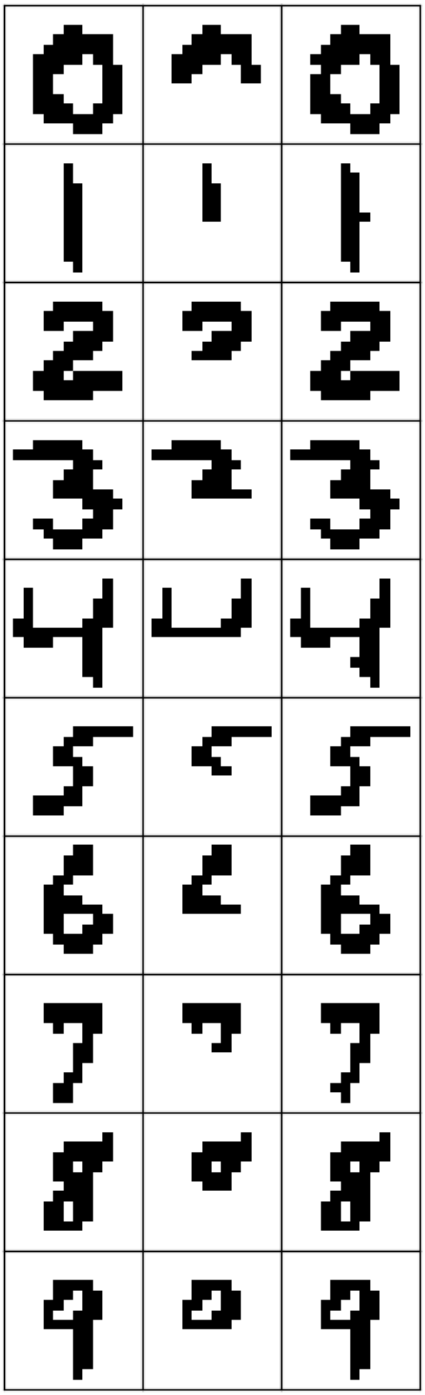

Figura 9 – Resultados obtidos para os dígitos de 0 a 9.
Fonte: Elaborada pelo autor.

Foram feitas 10 simulações para cada dígito e criado o gráfico abaixo mostrando a evolução da média e desvio padrão do fitness em função do número de gerações. Podemos observar a diminuição do desvio padrão de cada dígito na etapa final do algoritmo genético, indicando uma consistência nos resultados obtidos.

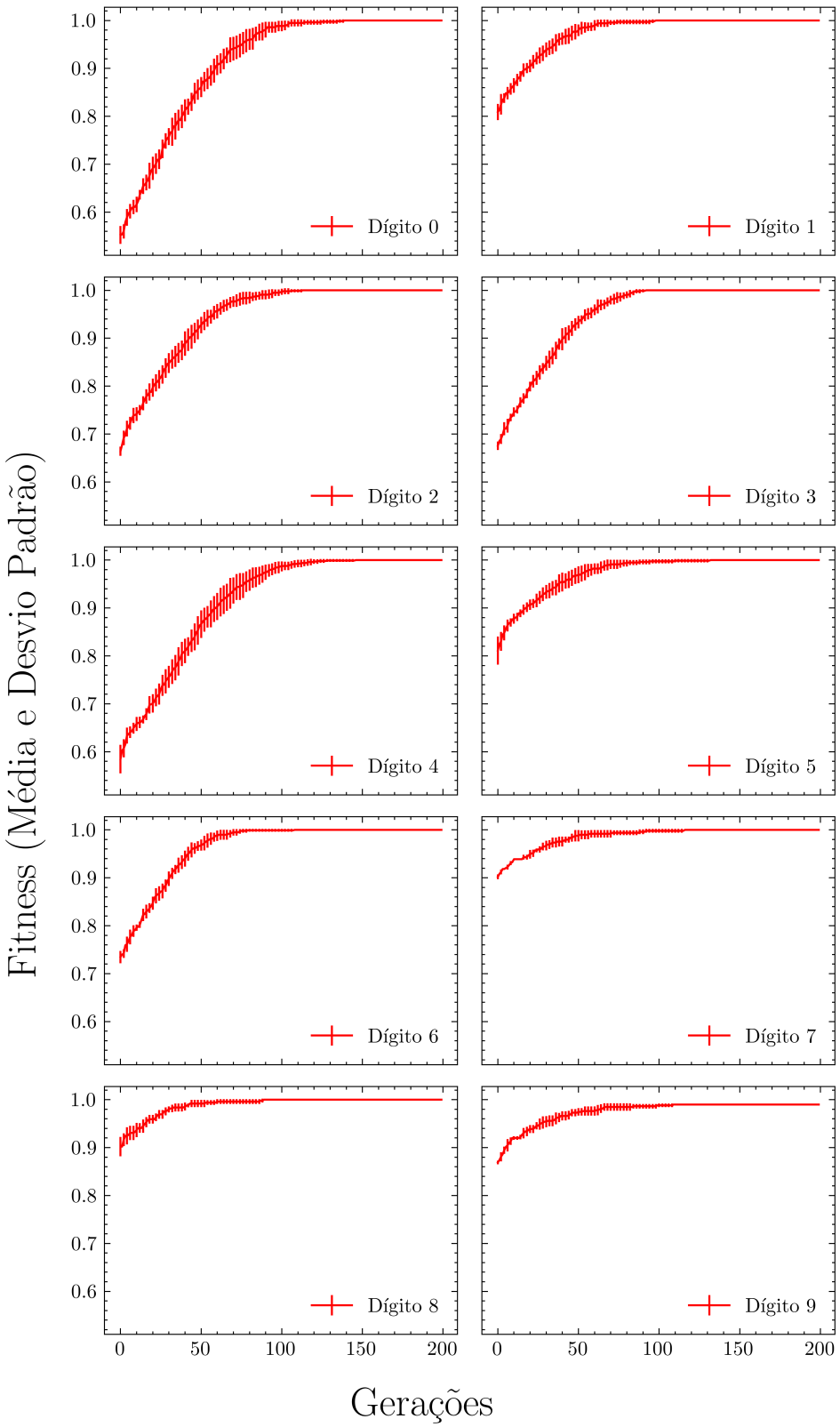

Figura 10 – Evolução do fitness para os diversos dígitos.
Fonte: Elaborada pelo autor.

4.3 Impacto da população inicial nos resultados obtidos

Cada indivíduo da população inicial é obtido multiplicando cada elemento do indivíduo sub-otimizado, que é a matriz de pesos sinápticos, por um valor de x, onde $x \in \{0, -1, 1\}$. Mas a probabilidade de escolha de x não é uniforme, neste trabalho foi utilizado o conceito de amostragem por importância. (7)

$$i = floor([rand()^{\beta} M]) + 1 \tag{4.1}$$

O parâmetro β permite que se faça um controle da probabilidade de se criar um número aleatório entre 1 e M, permitindo por exemplo que sejam gerados valores mais próximos de 1 ou de M. Substituindo $\beta = 0.65$ e $M = 3$ na equação, temos o histograma de frequência relativa obtido tomando o número de conexões sinápticas da rede como a quantidade de amostras, portanto foram geradas $N^2 = 196^2 = 38.416$ amostras aleatórias de valores entre 1, 2 e 3.

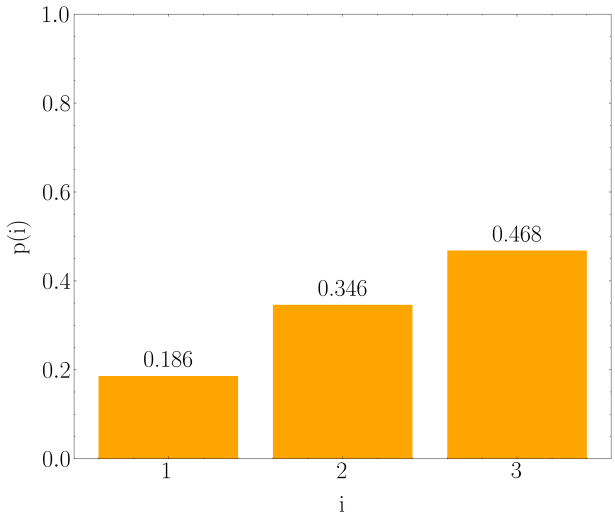

Figura 11 – Histograma obtido utilizando $\beta = 0.65$.
Fonte: Elaborada pelo autor.

Os valores de i representam os índices do vetor contendo os valores de x, no caso $[0, -1, 1]$ nesta ordem. A tabela abaixo foi criada a partir de alguns valores escolhidos de β para mostrar como esse parâmetro influência no desempenho do algoritmo genético.

Tabela 3 – Comparação das probabilidades entre os diversos valores de β.

β	Prob. conexão ser podada	Prob. conexão ser invertida	Prob. conexão mantida
0.2	0.4%	12.2%	87.4%
0.65	18.6%	34.6%	46.8%
2.2	60.8%	22.4%	16.8%

Fonte: Elaborada pelo autor.

O gráfico abaixo mostra a evolução do fitness do melhor indivíduo de cada geração quando se tenta recuperar o dígito 3 a partir de sua versão incompleta, utilizando os valores de β da tabela 3.

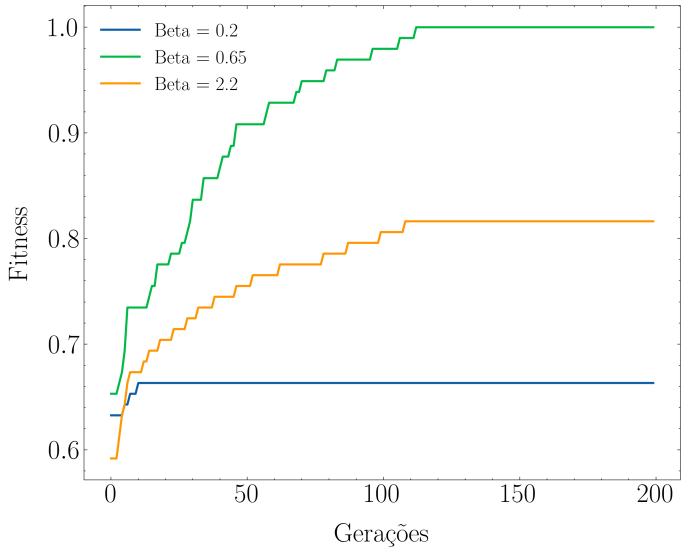

Figura 12 – Evolução do algoritmo genético para os diversos valores de β.
Fonte: Elaborada pelo autor.

Utilizando $\beta = 0.2$ o algoritmo genético converge prematuramente, ou seja, não atinge a solução ideal que corresponde ao fitness igual a 1. Isto ocorre devido a alta semelhança entre todos os indivíduos da população inicial, pois 87.4% das conexões sinápticas são mantidas, como consequência temos uma falta de diversidade na população inicial, assim a ação dos operadores genéticos de crossover e mutação quase não surtem efeitos, apenas no começo temos um pequeno ganho no valor do fitness, mas depois de uma certa geração o valor fica constante.

Para solucionarmos esse problema de convergência prematura, temos que aumentar a diversidade da população inicial, assim escolhendo $\beta = 2.2$, onde apenas 16.8% das conexões sinápticas são mantidas, podemos observar o funcionamento dos operadores genéticos, que vão aumentando o valor do fitness, mas ainda não é atingido a solução ideal, teríamos que rodar o algoritmo genético por mais gerações para observar se ele chega na solução ideal.

Escolhendo $\beta = 0.65$, temos um equilíbrio entre as probabilidades das conexões sinápticas serem podadas, invertidas ou mantidas o que resulta em uma convergência rápida em pouco mais de 100 gerações, o que torna este valor de β como sendo o ideal.

Portanto temos que a escolha de β permite que tenhamos um controle do tamanho dos pedaços da matriz de pesos sinápticos, que são distribuídos aleatoriamente entre os indivíduos da população inicial e que dependendo desse tamanho, os operadores genéticos são capazes de recuperar os padrões originais.

Podemos analisar a escolha de β observando graficamente os indivíduos, que são as matrizes de pesos sinápticos.

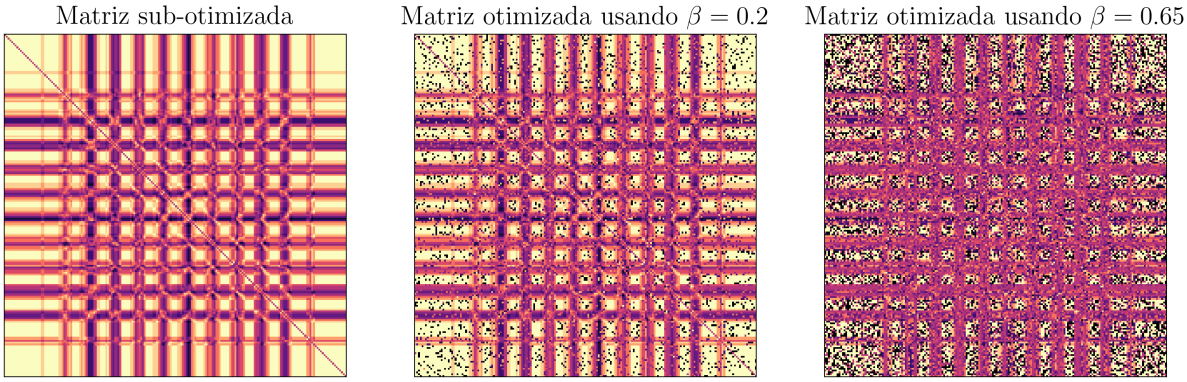

Figura 13 – Representação das matrizes sinápticas após a aplicação do algoritmo genético.
Fonte: Elaborada pelo autor.

Notamos que a matriz sub-otimizado é simétrica devido a regra de Hebb, e observando as matrizes após a aplicação do algoritmo genético, notamos a influência do β para os dois casos mostrados na figura, onde para $\beta = 0.65$ notamos uma maior alteração na matriz de pesos sinápticos. É possível também notar que durante a evolução do algoritmo genético a diagonal principal continua nula, situação necessária para a rede de Hopfield, pois os neurônios da rede não possuem retroalimentação. Para melhor observação foi aumentada a região inferior direita da última figura.

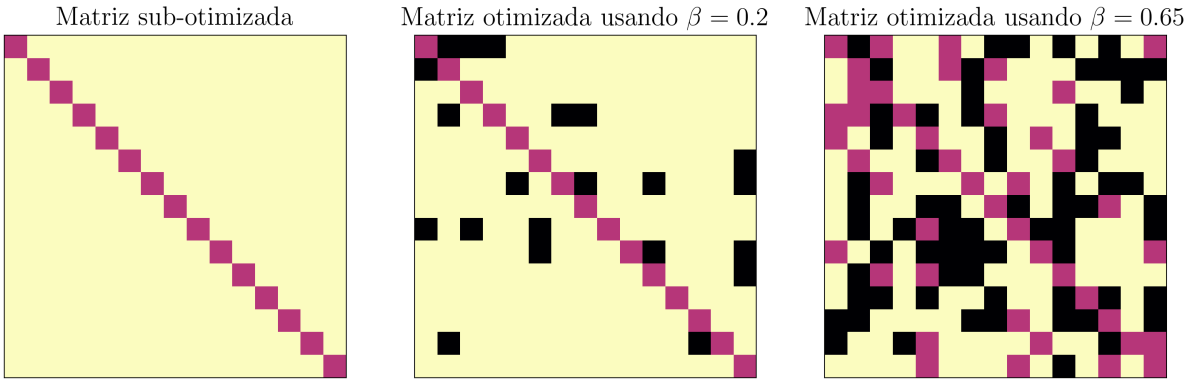

Figura 14 – A diagonal principal permanece nula após a aplicação do algoritmo genético.
Fonte: Elaborada pelo autor.

A diagonal principal permanece nula devido a implementação uniforme do operador genético de crossover, que impede que a operação de criação de novos filhos gere um com a diagonal principal não nula. Para o operador genético de mutação não é permitido alterar elementos que possuem a posição $i = j$.

4.4 Efeitos da paralelização no desempenho do algoritmo genético

Para que se possa quantificar o sucesso da paralelização é definida uma grandeza conhecida como *speedup*, que nada mais é que a razão entre o tempo de execução sequencial com o tempo de execução paralela do programa.

Os dados mostrados abaixo mostram o speedup obtido quando se tenta recuperar o dígito 6 a partir de sua versão incompleta. Os testes foram feitos em um computador com processador AMD Ryzen 5 1600 3.2GHz com 6 núcleos físicos (12 threads) e 16GB de memória RAM.

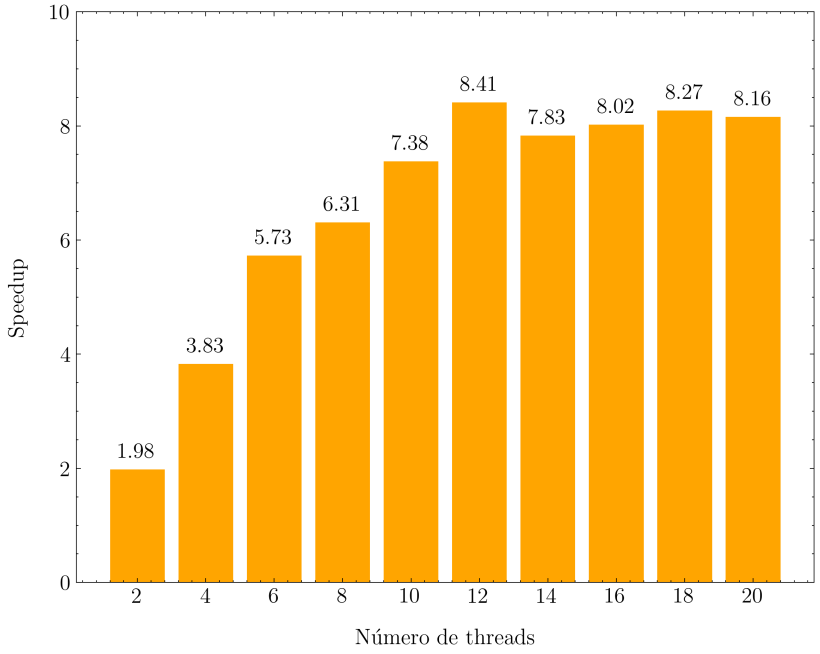

Figura 15 – Valores obtidos de speedup na recuperação do dígito 6.
Fonte: Elaborada pelo autor.

No gráfico percebemos a evolução do speedup conforme o aumento do número de threads até atingir o valor máximo de 8.41 para 12 threads. Observamos que o aumento do número de threads para um valor acima do que o processador possui, não aumenta o valor do speedup, pois neste caso como temos um número elevado de threads é necessário um gerenciamento na execução dessas threads, e esse gerenciamento consome tempo.

A principal vantagem na paralelização do algoritmo genético foi que permitiu que fossem feitos diversos testes na tentativa de encontrar o valor ideal para o parâmetro β. O procedimento adotado foi que após escolhido um valor qualquer de β, o programa era rodado para algumas dezenas de gerações e então era observado a evolução do fitness do melhor indivíduo de cada geração, assim foi possível ter uma noção se a escolha do parâmetro foi boa ou não.

5 CONCLUSÃO

Este trabalho fez uma análise do desempenho da rede de Hopfield atuando como memória associativa utilizando dígitos manuscritos da base de dados MNIST. Os experimentos mostraram que devido aos caracteres serem correlacionados, a rede não foi capaz de recuperar os padrões corretamente.

Utilizando o algoritmo genético de forma a otimizar as interconexões da rede, foi constatado a partir dos resultados obtidos, a eficácia do método, pois todos os 10 padrões utilizados no treinamento foram recuperados com no máximo 2 pixels de erros. Considerando que os padrões utilizados possuem um tamanho de 14x14 pixels, isto corresponde a uma porcentagem mínima de acerto de 99% no número de pixels corretamente recuperados.

Constatou-se que os parâmetros de inicialização do algoritmo genético tem forte influência nos resultados, principalmente a criação da população inicial. Os parâmetros que obtiveram melhores resultados durante a execução do algoritmo genético foram obtidos a partir de tentativa e erro.

A paralelização do algoritmo genético conseguiu atingir um speedup de 8.41, valor dentro do esperado considerando o computador na qual os testes foram feitos, mas a principal vantagem foi permitir que fossem realizados diversos testes com o intuito de encontrar o valor ideal para o parâmetro β, que permitisse uma rápida convergência do algoritmo genético. Para os tipos de dados usados neste trabalho o valor encontrado foi de $\beta = 0.65$.

Para trabalhos futuros, uma sugestão seria testar o método aqui desenvolvido para outros tipos de dados do mundo real, utilizar outras formas de perturbação que não sejam versões incompletas dos padrões armazenados, utilizar outros métodos de treinamento da rede de Hopfield e também a utilização de outras técnicas de paralelização, como por exemplo *MPI*.

REFERÊNCIAS

1 RAMSAUER, H. *et al.* **Hopfield networks is all you need**. 2021. Disponível em: https://arxiv.org/abs/2008.02217. Acesso em: 07 aug. 2021.

2 MENG, Z.; HU, Y.; ANCEY, C. Using a data driven approach to predict waves generated by gravity driven mass flows. **Water**, v. 12, n. 600, p. 1–18, 2020. Disponível em: https://www.mdpi.com/2073-4441/12/2/600. Acesso em: 07 jul. 2021.

3 OLIVEIRA, R. *et al.* A system based on artificial neural networks for automatic classification of hydro-generator stator windings partial discharges. **Journal of Microwaves, Optoelectronics Applications**, v. 16, n. 3, p. 628–645, Sept. 2017. Disponível em: https://www.scielo.br/j/jmoea/a/M7Cc4KCtC7KMtkxWVH55BVL. Acesso em: 05 jul. 2021.

4 HEBB, J. **The organization of behavior: a neuropsychological theory**. New York: John Wiley and Sons, Inc., 1949.

5 HOPFIELD, J. Neural networks and physical systems with emergent collective computational abilities. **Proceedings of the National Academy Science**, v. 79, n. 1, p. 2554–2558, 1982. Disponível em: https://www.researchgate.net/publication/16246447_Neural_Networks_and_Physical_Systems_with_Emergent_Collective_Computational_Abilities. Acesso em: 17 jun. 2021.

6 COSTA, L. F. **A compact guide to the Hopfield network**. 2021. Disponível em: https://www.researchgate.net/publication/348105865_A_Compact_Guide_to_The_Hopfield_Network_CDT-50. Acesso em: 16 jun. 2021.

7 COSTA, L. F. **A synthetic introduction to the genetic algorithm**. 2020. Disponível em: https://www.researchgate.net/publication/344016846_A_Synthetic_Introduction_to_the_Genetic_Algorithm_CDT-38. Acesso em: 19 jun. 2021.