# Projeto 2 - MC906 A

IGOR MATEUS OMOTE *, LUIZ EDUARDO CARTOLANO †

*†Engenharia de Computação - Graduação

*E-mail: i169819@students.ic.unicamp.br, †E-mail: l183012@students.ic.unicamp.br

***Resumo* – Este trabalho visa estudar a aplicação de algoritmos genéticos no mercado de ações (*stock marketing*) e *tradings*, encontrando os melhores parâmetros possíveis para serem aplicados em uma técnica de *trading* conhecida como *Regra Do Filtro*, de maneira menos custosa do que na maneira tradicional, a qual é feita a partir de um algoritmo guloso. O melhor cromossomo encontrado possibilitou um lucro de 168.1% e foi obtido em 1271.77 segundos, enquanto que, de acordo com resultados vistos na literatura, uma escolha gulosa não seria obtida em menos de uma hora. Outros resultados interessantes obtidos neste trabalho dizem respeito a observação do comportamento da solução à medida que alterávamos o tamanho da população ou as taxa de *crossover* e mutação.**

***Palavras-chave* – Inteligência artificial - Computação Evolutiva - Estratégias de *Trading***

## I. INTRODUÇÃO

Os algoritmos genéticos são uma metodologia muito comum nos ramos de *Inteligência Artificial* e *Machine Learning*. Eles são um método para resolver problemas de otimização restritos e não restritos, baseado na seleção natural, o processo que impulsiona a evolução biológica.

Esse tipo de algoritmo tem uma série de aplicações práticas e são vastamente utilizados nos dias de hoje. Seu uso não está restrito a uma área em específico, podendo ser aplicados nos mais diversos ramos. Alguns exemplos de aplicação são no *design* automotivo, na robótica, na otimização do roteamento de redes de telecomunicação e também no âmbito do mercado financeiro.

Para este trabalho será estudada a aplicação dos algoritmos genéticos no mercado de ações (*stock marketing*) e *tradings*. Este tipo de algoritmo tem sido aplicado em larga escala em problemas existentes no mercado financeiro[1]. Uma aplicação, por exemplo, é como encontrar os melhores valores de combinação de parâmetros para uma certa regra de negociação. Sabemos que em uma regra de negociação existem muitos parâmetros, quando tentamos encontrar o maior lucro, devemos testar a combinação de parâmetros um a um, o que caracteriza um algoritmo guloso, que apresenta alto custo computacional, como será visto em maior detalhe nas próximas seções.

Este trabalho encontra-se organizado da seguinte forma: a Seção II apresenta a maneira como o trabalho foi organizado a fim de solucionar o problema. A Seção III descreve os testes que foram feitos para validar a solução adotada e também as metodologias abordadas. Os resultados e suas respectivas análises são apresentados na Seção IV e por fim, as conclusões são apresentadas na Seção V.

Os códigos implementados para este trabalho encontram-se disponíveis em 2 (o *link* de acesso pode ser encontrado em 3, caso necessário).

## II. TRABALHO PROPOSTO

O trabalho proposto, que pode ser visto com mais detalhes em 4, solicita a implementação de algoritmos genéticos para solucionar um problema da literatura. Para resolver o problema este será especificado, bem como o modelo evolutivo adotado e os parâmetros que foram variados a fim de avaliar a solução. Para explicar o trabalho de maneira mais detalhada nas próximas subseções, serão abordados o problema, as ferramentas utilizadas para resolvê-lo e a organização do projeto.

### A. Problema

O problema proposto, consiste em otimizar uma situação do mundo real a partir do uso de algoritmos genéticos. Neste relatório será abordada a otimização de estratégias de *trading*, ou seja, a maximização do conjunto de parâmetros usados em uma estratégia de *trading*, a fim de maximizar o lucro obtido com a compra e venda de ações.

O objetivo do trabalho é implementar diferentes configurações de parâmetros do algoritmo evolutivo, a fim de avaliar sua *performance*, e resultado, sob diferentes condições de tamanho da população, critérios de parada e estratégias de *crossover*, *mutação* e *seleção*.

### B. Ferramentas utilizadas

A fim de realizar as implementações propostas utilizou-se como linguagem de programação a versão 3.7 do *Python*. Em conjunto com ela, usou-se a plataforma *Jupyter Notebook*, uma vez que com ela é possível acompanhar de maneira mais fácil os resultados dos algoritmos aplicados e também pois ela permite a execução de trechos específicos de código, o que garante maior flexibilidade no uso dos diversos algoritmos usados.

Outro detalhe importante das ferramentas utilizadas foi o uso de dados das ações obtidos pela plataforma *Reuters Eikon*[1]. A partir dela foi possível obter o histórico de flutuação de preços da ação escolhida em um determinado período.

[1] https://tinyurl.com/yyzasfx9

### C. Organização do projeto

O projeto foi estruturado de modo que ele possui dois arquivos principais que implementam a modelagem do problema, a estratégia de *trading* utilizada e os testes feitos com o algoritmo genético. O arquivo *greedy_filter.py* implementa a estratégia gulosa que realiza a compra e venda das ações com base nos parâmetros fornecidos pelos cromossomos. Já o algoritmo *genetic_algorithm.ipynb* implementa os métodos associados a estratégia evolutiva (critério de parada, mutação, *crossover* e seleção). Além do conjunto de testes que foram realizados.

## III. Materiais e Métodos

Nesta seção iremos explicar com mais detalhes a modelagem do problema, o modelo evolutivo adotado, os testes realizados para analisar a corretude dos algoritmos aplicados e também será comentado sobre eventuais restrições apresentadas nas soluções. Também será dada uma breve introdução sobre os conceitos mais importantes relacionados a este trabalho.

### A. Modelagem do Problema

Nesta subseção serão dados mais detalhes de como o problema foi abordado, ou seja, como foi feita a escolha dos parâmetros usados na estratégia de *trading*. Para tal, será explicado, em detalhes, cada uma das configurações adotadas associando elas com uma característica do algoritmo genético.

*1) Gene:* Os genes representam cada um dos parâmetros da estratégia, sendo eles: a porcentagem de crescimento(x), a quantidade de dias de *hold*(h), a quantidade de dias de *delay*(d) e a quantidade de dias anteriores(p). Cada um deles será visto com mais detalhes na Seção III-B.

*2) Cromossomo:* Os cromossomos serão formados pelo conjunto de todos os parâmetros, ou seja, o conjunto de genes.

*3) População:* A população será formada por um conjunto de cromossomos, ela é inicializada de maneira randômica e seu tamanho inicial será variado a fim de analisar a convergência do algoritmo sob diferentes situações.

*4) Mutação e Crossover:* Mutação e *crossover* foram as técnicas de variação usadas para gerar novos cromossomos. A primeira técnica de mutação utilizada varia os valores de algum gene escolhido aleatoriamente para um valor também aleatório que esteja contido em uma faixa estipulada pré determinada. Uma segunda técnica de mutação escolhida foi realizar aleatorizar os valores de um intervalo que vai de um gene x até um y, sendo $x, y > 0$ e $y >= x$.

Para o *crossover* também foram usadas duas técnicas diferentes: com um ponto de *crossover* e com dois pontos. Tais estratégias necessitam do *input* de dois cromossomos, os pontos de *crossover* são escolhidos aleatoriamente e com essas informações, a técnica gera dois cromossomos novos. Em linhas gerais, para o algoritmo de *crossover* de um ponto, basta selecionar um inteiro aleatório N menor que o tamanho do cromossomo, e, para o primeiro cromossomo filho, seleciona-se os N primeiros genes de um dos cromossomos-pai e os restantes do outro cromossomo pai, para o segundo cromossomo-filho, os demais genes restantes. Da mesma forma, o *crossover* de dois pontos aleatoriza dois inteiros N e M, o primeiro cromossomo-filho receberá os primeiros N genes do primeiro cromossomo-pai, os M - N genes do segundo cromossomo-pai e os outros M genes do primeiro cromossomo-pai, analogamente o inverso ocorre para o segundo cromossomo-filho.

*5) Seleção:* Para a implementação do projeto foram usadas duas técnicas de *Seleção*. A primeira delas selecionava os *N* melhores indivíduos de uma dada população para dar continuidade ao algoritmo. Já a segunda técnica foi o método da *Roleta*, que seleciona indivíduos aleatoriamente, proporcionando chances de reprodução aos membros mais aptos, visto que estes possuem maior probabilidade de serem escolhidos. Para evitar que *super indivíduos* se sobressaíssem foi feita uma normalização linear dos valores de *fitness*.

*6) Critério de Parada:* Os *Critérios de Parada* escolhidos foram inteiramente relacionados a convergência da função *fitness*, e as variações feitas sobre eles dizem respeito a margem usada para considerarmos se o *fitness* de uma população estava ou não convergindo. Além disso, determinou-se um número mínimo de iterações que deveriam acontecer para se tentar prevenir a convergência prematura.

### B. Introdução Teórica

*1) Algoritmos Genéticos:* Os *Algoritmos Genéticos* são uma função heurística para otimização, no qual o extremo da função não pode ser estabelecido analiticamente. Uma população de soluções potenciais é refinada iterativamente empregando uma estratégia inspirada pela evolução *darwinista* ou seleção natural. Algoritmos genéticos promovem a "sobrevivência do mais apto".

Para este trabalho representou-se os parâmetros de uma regra de *trading* com um vetor que serão nossos *cromossomos*, cada cromossomo, em especial, é formado de *genes* e todos os *cromossomos* formam uma *população*. Geralmente, as operações genéticas incluem: *crossover*, *mutação* e *seleção*.

O *Algoritmo Genético* pode ser descrito, de forma simplificada, pelos passos vistos no Algoritmo 1.

*2) Trading:* O *Trade* é um conceito econômico básico que envolve a compra e venda de bens e serviços. No âmbito do mercado financeiro, a negociação refere-se à compra e venda de títulos, como a compra de ações no pregão da *Bolsa de Nova York (NYSE)*. No mercado de ações, quando os corretores(ou revendedores) querem comprar ou vender uma ação, alguns deles usarão o auxílio de uma regra técnica de negociação. Essas podem ser definidas como "a ciência de registrar a história real da negociação (mudanças de preço, volume de transações, etc.) em uma determinada ação ou nas *Médias* e deduzir dessa história a provável tendência futura."

Desde o século passado, tem surgido muitas regras para direcionar as *tradings*, tais como: Regras de Filtro, Média Móvel, Suporte e Resistência, Retorno Anormal, etc [5].

A estratégia de *trading* utilizada foi uma variação da *Regra de Filtro - Filter Rule*, desenvolvida por Alexander, em 1961, e que foi explicada em mais detalhes em 6. Sua definição é dada da seguinte maneira, compre e segure um título até que seu

preço caia x por cento em relação a uma alta subsequente, momento no qual esse título deve ser vendido. Essa nova posição deve ser mantida até que os preços da nova ação aumentem pelo menos x por cento com relação a uma baixa, nesse momento deve-se comprá-las novamente. Movimentos menores que x por cento devem ser ignorados.

A *Regra de Filtro* faz uso de quatro parâmetros na sua aplicação, que são:

- X - PORCENTAGEM DE COMPARAÇÃO: o parâmetro X indica quantos porcento o valor da ação deve ter aumentado(ou diminuído) com relação ao último pico(ou vale) para que seja emitido um sinal de compra(ou venda).
- H - DIAS EM HOLD: o parâmetro H indica quantos dias deve-se ficar sem tomar qualquer atitude após a emissão de um sinal de compra(ou venda). Por exemplo, se foi emitido um sinal de compra no dia *01/05* e o *hold* previsto é de cinco dias, então até o dia *06/05* nenhum novo sinal - seja ele qual for - será emitido pelo algoritmo.
- D - DIAS DE DELAY: o parâmetro D existe para dar uma certa robustez ao algoritmo, isto é, a ideia de sua existência é evitar que o algoritmo tome decisões ruins por conta de alguns ruídos que possam existir nos valores analisados. Para tal, ele é uma quantidade de dias, após o período de *hold* no qual se analisa os valores a fim de encontrar um novo sinal que seja válido (ou seja, um sinal de venda caso você possua ações, ou um de compra caso você não as possua).
- P - QUANTIDADE DIAS ANTERIORES: o parâmetro P indica a quantidade dias que será usada na comparação dos valores, ou seja, até quantos dias antes da data atual deve se procurar por picos(ou vales) nos dados que estão sendo analisados.

Um detalhe importante é que para este trabalho, adotou-se a seguinte política, todo o *budget* disponível será gasto na compra de ações quando tal sinal for emitido.

### C. *Testes de Corretude e* Performance

A fim de avaliar o comportamento das técnicas implementadas (métodos de *crossover*, mutação e seleção), abordou-se uma estratégia na qual foram criadas duas estratégias evolutivas, cada uma delas usando um método de *Seleção* diferente.

Para cada uma das estratégias, variou-se as taxas de *crossover*, taxa de mutação e as técnicas de *crossover* e mutação, totalizando um conjunto de nove testes para uma dada população, um *budget* inicial e um certo delta usado como critério de convergência.

Por fim, também criou-se um teste visando analisar a *performance* do algoritmo a medida que se aumentava o tamanho da população. Para tal, fixou-se os valores de *budget*, delta de convergência e adotou-se as taxas de mutação e *crossover* que, com suas respectivas técnicas, apresentaram o melhor resultado para o primeiro conjunto de testes. Com esses parâmetros fixados, variou-se a população com valores de dez até cem cromossomos.

Os dois últimos testes feitos com o algoritmo foram aplicar as técnicas criadas para os valores da moeda virtual *Bitcoin* [2] com relação ao dólar no intervalo dos últimos dois anos. O objetivo desse teste foi avaliar o algoritmo sob condições de alta variação, já que o valor da *criptomoeda* sofreu grande flutuação nos últimos anos. E também avaliar o lucro que seria obtido caso a compra de certas ações fosse feita quando elas estivessem com o menor valor (mínimo global) e a venda no seu maior valor (máximo global).

### D. *Restrições da Solução*

A implementação tem uma restrição significativa no que diz respeito a maneira como o *budget* inicial é gasto. Isso porque, diferente do mundo real, adotou-se uma estratégia na qual todo o dinheiro que se possuía era gasto em uma única ação, quando na verdade, o ideal seria a possibilidade de se investir em um conjunto heterogêneo de ações. Além disso, outra restrição apresentada pela técnica de *trading* adotada foi que o algoritmo é seguido estritamente na abordagem, portanto, algumas situações que deveriam ser corrigidas pelo *trader* são ignoradas, podendo ocasionar perdas que não aconteceriam normalmente.

Ademais, o algoritmo desenvolvido apenas trabalha com uma única ação, perdendo casos quando, para dado um período de tempo, identifica uma oportunidade melhor em outra ação mas não pode executá-la.

Por fim, restringiu-se o domínio de decisão dos genes para valores que fazem sentido em um intervalo de dois anos, em determinadas análises de *trade* isso pode não levar a um resultado ótimo.

## IV. Resultados e Discussão

Nesta seção primeiro serão apresentados todos os resultados obtidos durante a execução da solução implementada, incluindo tabelas e gráficos gerados. Uma vez introduzidos, serão discutidos e comparados.

### A. *Resultados*

Os resultados obtidos para a estratégia evolutiva usando o método de seleção por elitismo podem ser observados na Figura 1. Já os resultados para a estratégia que fez uso do método de seleção da roleta estão apresentados na Figura 2.

Para a estratégia que fez uso do método de seleção por elitismo, o melhor cromossomo foi encontrado no Teste #6, e é dado por $[0.031, 15, 19, 402]$, este apresentou um *fitness* de $166.9\%$. Já o pior cromossomo foi obtido no Teste #9, e é dado pelo cromossomo $[0.037, 17, 10, 235]$, seu *fitness* foi de $-6.1\%$. A melhor média de resultados de *fitness* foi obtida no Teste #2, com um valor de $(0.48 \pm 0.09)$. O Teste #7 foi o mais rápido levando 362.72 segundos para convergir enquanto que o Teste #8 foi o mais lento precisando de 2082.23 segundos para terminar.

Para a estratégia que fez uso do método de seleção da roleta, o melhor cromossomo foi encontrado no Teste #1, e é dado por $[0.029, 16, 10, 339]$, este apresentou um *fitness* de

[2] https://bitcoin.org/en/

$168.1\%$. Já o pior cromossomo foi obtido no Teste #9, e é dado pelo cromossomo $[0.025, 21, 5, 326]$, seu *fitness* foi de $-3.3\%$. A melhor média de resultados de *fitness* foi obtida no Teste #4, com um valor de $(0.46 \pm 0.45)$. Os Testes #4 e #9 foram os mais rápidos levando 864.27 segundos para convergir enquanto que o Teste #2 foi o mais lento precisando de 6348.25 segundos para terminar.

Na Figura 3 é possível observar o gráfico que apresenta o comportamento da solução à medida que aumentou-se o tamanho da população, no caso, um comportamento quase linear. Já a Figura 4 mostra o comportamento do tempo conforme variou-se a a taxa de mutação, como é possível perceber não há qualquer relação forte entre ambos os valores. Enquanto que a Figura 5 mostra a variação do tempo à medida que aumentou-se a taxa de *crossover*, uma relação muita próxima de um comportamento polinomial de grau 2. Estes resultados foram obtidos para a técnica de seleção da roleta, para o elitismo obteve-se resultados muito próximos. Por exemplo, manteve-se a linearidade a medida que aumentava-se a população como nos mostra a Figura 6, assim como manteve-se a falta de correlação para a taxa de mutação, como pode ser visto na Figura 4. Por fim, o comportamento para o crescimento da taxa de *crossover*, como mostra a Figura 8, nos lembra a parte positiva de um polinômio de grau dois.

### B. Discussão

Ao comparar o melhor resultado obtido fazendo uso do método de seleção por *elitismo* com o da *roleta*, observa-se que o segundo leva uma ligeira vantagem. É possível observar que este obtém o melhor cromossomo, quando comparadas as populações finais em oito dos nove testes feitos. O resultado condiz com o esperado pela literatura. Visto que, o elitismo depende da ”sorte”para obter uma boa população inicial ou de obter bons cromossomos pós mutação. Por outro lado, o algoritmo da roleta é probabilístico, logo é levemente mais independente.

O *trade-off* existente para o algoritmo da roleta, por outro lado, é que uma vez que escolhe seus elementos baseado em probabilidade, é comum a seleção de uma população mais heterogênea, ou seja, com elementos mais dispersos entre si. Duas consequências que podemos observar dessa característica é o maior tempo que a estratégia evolutiva precisava até convergir e, também, o maior desvio padrão da população final quando comparado com o método do elitismo.

Outro ponto que é possível de se observar ao se comparar os dois métodos de seleção é que o *elitismo* possui uma maior constância de resultados, isto é, além de ter uma média maior para os dados, o desvio padrão de suas amostras é consideravelmente menor quando comparado ao método da roleta. Nesse aspecto é interessante notar que ao combinar a seleção por *elitismo* com uma alta taxa de *crossover* obteve-se a população com menor desvio nos dados.

Além disso, também foi perceptível o comportamento esperado da solução conforme aumentava-se o tamanho da população, visto que, para ambos os métodos o tempo cresceu quase que linearmente com relação a população. Além disso, analisar o comportamento do tempo à medida que aumentava-se a taxa de mutação também trouxe resultados esperados, visto que praticamente não houve relação entre ambos. Contudo os testes com alta taxa de *crossover* (testes #2, #5, #8), deveriam convergir mais rapidamente do que os com baixa taxa de *crossover*, uma vez que em ambos métodos de seleção, haveria maior probabilidade da perpetuação dos genes com maior *fitness*, até que estes dominassem toda a população. A hipótese que para explicar os testes mencionados, é de que a população convergiu no mínimo de iterações e o tempo maior é devido ao esforço necessário para calcular o *fitness* de cada um dos novos cromossomos.

Alguns resultados, contudo, fugiram do esperado teórico: uma vez que o elitismo mantém os melhores resultados e perpetuam parte de seus genes no *crossover* com mais ”facilidade”, era esperado que a média dos *fitness* fosse muito mais próxima do maior valor encontrado para este do que ocorreu nos testes feitos.

Por fim, dois resultados satisfatórios que merecem ser comentados são o comportamento da solução para preços com alta taxa de variação, como no caso do *Bitcoin* e a comparação do lucro que foi obtido pela estratégia evolutiva com o que seria obtido caso a compra das ações fosse feita no momento de mínimo global dos preços e a venda no de máximo global. Para o primeiro, obteve-se resultados estarrecedores, com um lucro, no melhor cromossomo de $1591\%$. Já para o segundo, o lucro que seria obtido na situação descrita seria de $44.28\%$, enquanto que o lucro obtido pela estratégia evolutiva foi, no melhor caso, de $168.1\%$ e na média, como visto nas Figuras 1 e 2, algo em torno de $40\%$, o que reitera o sucesso da abordagem adotada. Ambos mostram a robustez e qualidade da técnica implementada inclusive em situações críticas.

## V. Conclusões

De modo geral, o trabalho apresentou resultados satisfatórios, especialmente quando levado em consideração que o objetivo deste era introduzir os alunos aos conceitos de algoritmos genéticos e computação evolutiva, sendo, ao fim deste, capaz de entender e analisar os algoritmos implementados de maneira não simplista e, principalmente, criar no aluno a capacidade crítica para modelar aplicações dos algoritmos estudados para problemas reais. Vale ainda ressaltar que os resultados obtidos pela aplicação da solução ao problema proposto foram significativamente positivos.

Dos métodos e materiais usados e vistos em detalhes na Seção III, ficam como pontos positivos do trabalho, a implementação manual das técnicas de seleção, *crossover* e mutação, e também, das estratégias evolutivas, trazendo um acúmulo muito maior de conhecimento do que caso fossem apenas realizadas chamadas para uma biblioteca externa. Outro ponto positivo foi a bateria de testes a qual o projeto foi submetido, pois ela permitiu a geração de dados importantes para serem analisados, permitindo observações mais profundas dos algoritmos.

Pontos fracos do trabalho, que precisam ser melhorados em iterações futuras, estão em sua maioria relacionados a

simplicidade de alguns modelos adotados para a modelagem do problema.

+————————-+

## REFERÊNCIAS


[1] S.-h. Chen, "Genetic algorithms and genetic programming in computational finance." *J. Artificial Societies and Social Simulation*, vol. 7, 01 2004. 1

[2] L. Cartolano and I. Omote. Códigos da solução. [Online]. Available: https://github.com/luizcartolano2/mc906/tree/master/projeto2 1

[3] ——. Link de acesso ao github. [Online]. Available: https://github.com/luizcartolano2/mc906/invitations 1

[4] E. Colombini. Enunciado projeto 1. [Online]. Available: http://www.ic.unicamp.br/~esther/teaching/2019s1/mc906/P2.pdf 1

[5] R. D. Edwards, J. Magee, and W. C. Bassetti, *Technical analysis of stock trends*. CRC press, 2018. 2

[6] Filter rule - alexander. [Online]. Available: http://www.farmdoc.illinois.edu/marketing/agmas/reports/04_04/Tables.htm 2


## ANEXOS

| Teste | Parâmetros | Melhor Cromossomo | Melhor fitness | [pior cromossomo] | Pior fitness | Média +- desvio | Tempo Execução [s] |
|---|---|---|---|---|---|---|---|
| #1 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.3<br>Mutação → mutation<br>Taxa Mutação → 0.3 | [0,011, 23, 15, 176] | 1,258 | [0.015, 14, 5, 407] | 0,013 | 0,29 ± 0,06 | 570,43 |
| #2 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.8<br>Mutação → mutation<br>Taxa Mutação → 0.3 | [0,080, 29, 3, 121] | 0,904 | [0.088, 7, 8, 465] | 0,026 | 0,48 ± 0,09 | 1631,87 |
| #3 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.3<br>Mutação → mutation<br>Taxa Mutação → 0.8 | [0,025, 19, 22, 291] | 1,258 | [0.085, 3, 4, 153] | 0,026 | 0,22 ± 0,03 | 583,26 |
| #4 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover2<br>Taxa Crossover → 0.3<br>Mutação → mutation<br>Taxa Mutação → 0.3 | [0,026, 25, 20, 281] | 1,218 | [0.027, 5, 12, 259] | 0,018 | 0,28 ± 0,06 | 580,42 |
| #5 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover2<br>Taxa Crossover → 0.8<br>Mutação → mutation<br>Taxa Mutação → 0.3 | [0,080, 12, 1, 236] | 0,904 | [0.087, 27, 4, 185] | 0,026 | 0,45 ± 0,09 | 1936,81 |
| #6 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover2<br>Taxa Crossover → 0.3<br>Mutação → mutation<br>Taxa Mutação → 0.8 | [0,031, 15, 19, 402] | 1,669 | [0.044, 10, 10, 425] | 0,017 | 0,26 ± 0,05 | 604,51 |
| #7 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.3<br>Mutação → mutation_v2<br>Taxa Mutação → 0.3 | [0,021, 20, 9, 331] | 1,190 | [0.088, 23, 8, 143] | 0,026 | 0,26 ± 0,04 | 362,72 |
| #8 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.8<br>Mutação → mutation_v2<br>Taxa Mutação → 0.3 | [0,080, 5, 3, 235] | 0,904 | [0.084, 23, 24, 104] | 0,026 | 0,45 ± 0,09 | 2082,23 |
| #9 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.3<br>Mutação → mutation_v2<br>Taxa Mutação → 0.8 | [0,040, 25, 5, 411] | 1,410 | [0.037, 17, 10, 235] | -0,061 | 0,40 ± 0,07 | 471,85 |

Figura 1. Resultado dos testes executados para o método de seleção por elitismo.

```
populacao := InicializaPopulacao() ;
while not condicaoDeParada(populacao) do
    parents[1..n] := populacao();
    descendentes[1] := crossover(parents);
    descendentes[2] := mutacao(parents);
    populacao := selecao(parents+descendentes, tamanho)
end
```

**Algorithm 1:** Pseudocódigo de um Algoritmo Genético

| Teste | Parâmetros | Melhor Cromossomo | Melhor fitness | [pior cromossomo] | Pior fitness | Média +- desvio | Tempo Execução [s] |
|---|---|---|---|---|---|---|---|
| #1 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.3<br>Mutação → mutation<br>Taxa Mutação → 0.3 | [0.029, 16, 10, 339] | 1,681 | [0.090, 24, 3, 304] | 0,000 | 0.36 ± 0.41 | 1271,77 |
| #2 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.8<br>Mutação → mutation<br>Taxa Mutação → 0.3 | [0.034, 16, 5, 328] | 1.478 | [0.092, 23, 5, 276] | 0,000 | 0.37 ± 0.39 | 6348,25 |
| #3 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.3<br>Mutação → mutation<br>Taxa Mutação → 0.8 | [0.054, 2, 11, 41] | 1.476 | [0.092, 18, 25, 435] | 0,000 | 0.39 ± 0.41 | 1398,42 |
| #4 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover2<br>Taxa Crossover → 0.3<br>Mutação → mutation<br>Taxa Mutação → 0.3 | [0.036, 26, 15, 313] | 1.478 | [0.097, 11, 18, 34] | 0,000 | 0.46 ± 0.45 | 864,27 |
| #5 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover2<br>Taxa Crossover → 0.8<br>Mutação → mutation<br>Taxa Mutação → 0.3 | [0.020, 1, 15, 400] | 1.447 | [0.094, 13, 5, 333] | 0,000 | 0.33 ± 0.38 | 4823,03 |
| #6 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover2<br>Taxa Crossover → 0.3<br>Mutação → mutation<br>Taxa Mutação → 0.8 | [0.017, 13, 7, 67] | 1.466 | [0.027, 16, 7, 315] | -0,028 | 0.38 ± 0.40 | 1780,06 |
| #7 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.3<br>Mutação → mutation_v2<br>Taxa Mutação → 0.3 | [0.051, 8, 28, 45] | 1.406 | [0.065, 20, 9, 323] | 0,031 | 0.34 ± 0.38 | 2130,24 |
| #8 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.8<br>Mutação → mutation_v2<br>Taxa Mutação → 0.3 | [0.011, 14, 1, 482] | 1,649 | [0.050, 27, 2, 187] | 0,009 | 0.40 ± 0.45 | 3403,16 |
| #9 | População → 100<br>Budget → 10.000<br>Delta → 0.01<br>Crossover → crossover<br>Taxa Crossover → 0.3<br>Mutação → mutation_v2<br>Taxa Mutação → 0.8 | [0.025, 21, 5, 326] | 1,454 | [0.069, 10, 8, 436] | -0,033 | 0.39 ± 0.41 | 864,27 |

Figura 2. Resultado dos testes executados para o método de seleção da roleta.

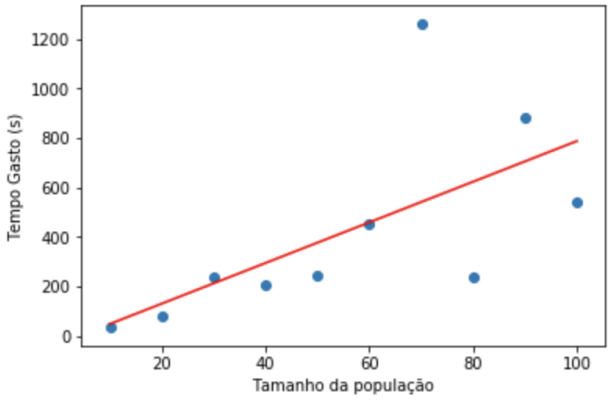


Figura 3. Comportamento do tempo a medida que aumentava-se a população para o método de seleção da roleta.

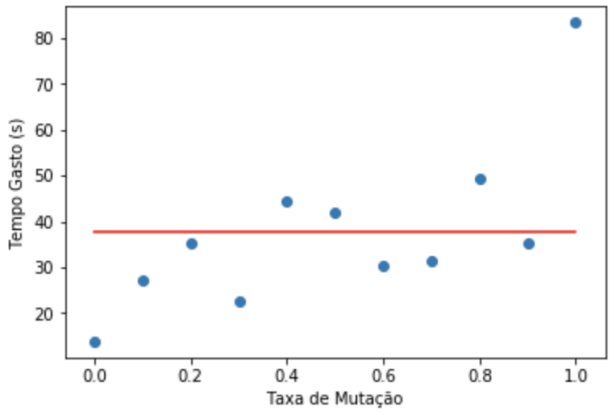


Figura 4. Comportamento do tempo a medida que aumentava-se a taxa de mutação para o método de seleção da roleta.

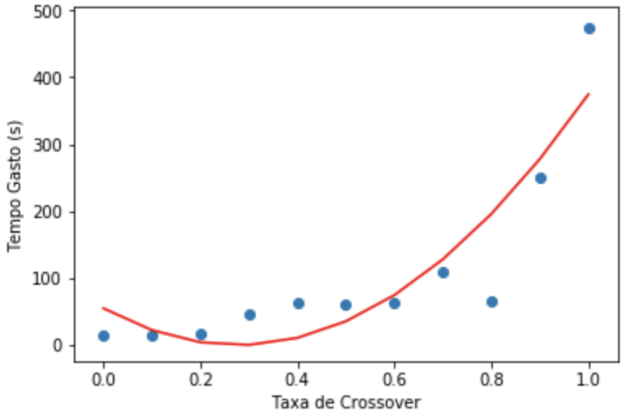


Figura 5. Comportamento do tempo a medida que aumentava-se a taxa de crossover para o método de seleção da roleta.

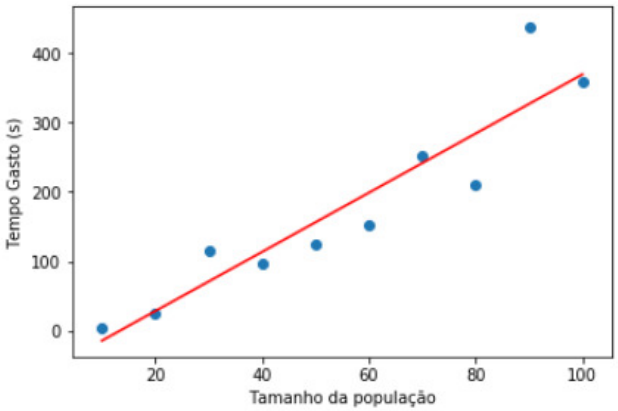


Figura 6. Comportamento do tempo a medida que aumentava-se a população para o método de seleção por elitismo.

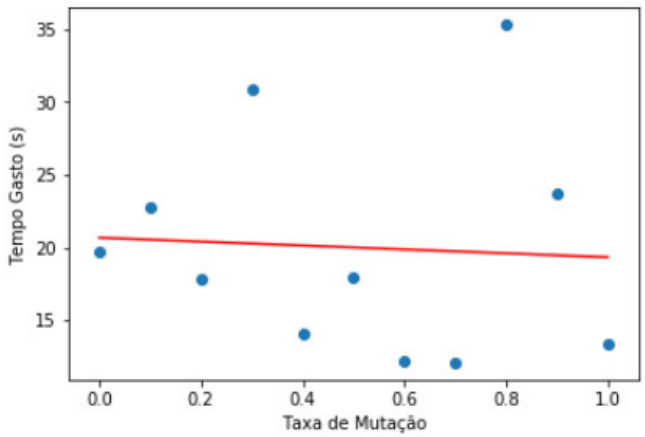


Figura 7. Comportamento do tempo a medida que aumentava-se a taxa de mutação para o método de seleção por elitismo.

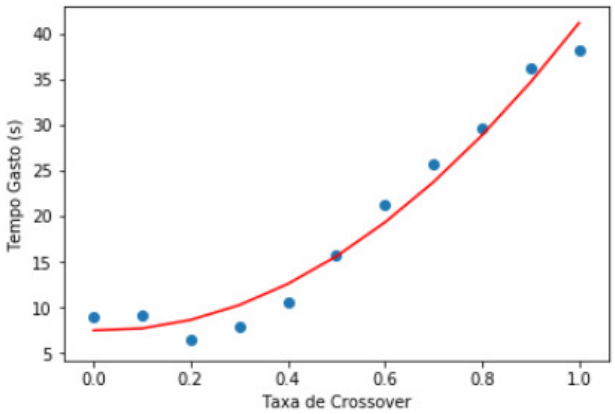


Figura 8. Comportamento do tempo a medida que aumentava-se a taxa de crossover para o método de seleção por elitismo.