BACHAREL EM CIÊNCIA DA COMPUTAÇÃO

ANÁLISE DE ALGORITMO EVOLUTIVO PARA O PROBLEMA DE EMPARELHAMENTO DE CARDINALIDADE MÁXIMA

ATHOS JOSÉ DE ARAÚJO

Rio Verde, GO

2023

INSTITUTO FEDERAL GOIANO - CAMPUS RIO VERDE
BACHAREL EM CIÊNCIA DA COMPUTAÇÃO

ANÁLISE DE ALGORITMO EVOLUTIVO PARA O PROBLEMA DE EMPARELHAMENTO DE CARDINALIDADE MÁXIMA

ATHOS JOSÉ DE ARAÚJO

Trabalho de Conclusão de Curso apresentado ao Instituto Federal Goiano - Campus Rio Verde, como requisito parcial para a obtenção do Grau de Bacharel em Ciência da Computação.

Orientador: Prof. ANDRÉ DA CUNHA RIBEIRO

Rio Verde, GO

06, 2023

Sistema desenvolvido pelo ICMC/USP
Dados Internacionais de Catalogação na Publicação (CIP)
Sistema Integrado de Bibliotecas - Instituto Federal Goiano

Araújo, Athos José de
A663a ANÁLISE DE ALGORITMO EVOLUTIVO PARA O PROBLEMA DE EMPARELHAMENTO DE CARDINALIDADE MÁXIMA / Athos José de Araújo; orientador André da Cunha Ribeiro. -- Rio Verde, 2023.
26 p.

TCC (Graduação em Bacharelado em Ciência da Computação) -- Instituto Federal Goiano, Campus Rio Verde, 2023.

1. Algoritmo Evolutivo. 2. Grafo. 3. (1+1) EE. 4. Emparelhamento. I. Ribeiro, André da Cunha, orient. II. Título.

Responsável: Johnathan Pereira Alves Diniz - Bibliotecário-Documentalista CRB-1 n°2376

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA GOIANO

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR PRODUÇÕES TÉCNICO-CIENTÍFICAS NO REPOSITÓRIO INSTITUCIONAL DO IF GOIANO

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia Goiano, a disponibilizar gratuitamente o documento no Repositório Institucional do IF Goiano (RIIF Goiano), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IF Goiano.

Identificação da Produção Técnico-Científica (assinale com X)

[ ] Tese

[ ] Dissertação

[ ] Monografia – Especialização

[ ] Artigo - Especialização

[x] TCC - Graduação

[ ] Artigo Científico

[ ] Capítulo de Livro

[ ] Livro

[ ] Trabalho Apresentado em Evento

[ ] Produção técnica. Qual: _____

Nome Completo do Autor: Athos José de Araújo

Matrícula: 2016102192010277

Título do Trabalho: ANÁLISE DE ALGORITMO EVOLUTIVO PARA O PROBLEMA DE EMPARELHAMENTO DE CARDINALIDADE MÁXIMA

Restrições de Acesso ao Documento [Preenchimento obrigatório]

Documento confidencial: [x] Não [ ] Sim, justifique: _____

Informe a data que poderá ser disponibilizado no RIIF Goiano: 17/08/2023

O documento está sujeito a registro de patente? [ ] Sim [x] Não

O documento pode vir a ser publicado como livro? [ ] Sim [x] Não

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

1. O documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;

2. Obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia Goiano os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;

3. Cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia Goiano.

Rio Verde, 17 de agosto de 2023

Athos José de Araújo

Assinado eletronicamente pelo o Autor e/ou Detentor dos Direitos Autorais

Ciente e de acordo:

André da Cunha Ribeiro

Assinatura eletrônica do(a) orientador(a)

Documento assinado eletronicamente por:

- **Athos José de Araújo**, **2016102192010277 - Discente**, em 17/08/2023 13:54:09.
- **Andre da Cunha Ribeiro**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 17/08/2023 13:52:50.

Este documento foi emitido pelo SUAP em 17/08/2023. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifgoiano.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 522620
Código de Autenticação: 8da794afda

INSTITUTO FEDERAL GOIANO
Campus Rio Verde
Rodovia Sul Goiana, Km 01, Zona Rural, 01, Zona Rural, RIO VERDE / GO, CEP 75901-970
(64) 3624-1000

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA GOIANO

Ata nº 25/2023 - GEPTNM-RV/DE-RV/CMPRV/IFGOIANO

ATA DE DEFESA DE TRABALHO DE CURSO

Ao(s) 04 dia(s) do mês de julho de 2023, às 15 horas, reuniu-se a banca examinadora composta pelos docentes: Dr. André da Cunha Ribeiro (orientador), Dr. Douglas Cedrim Oliveira (membro), Dr. Márcio Antônio Ferreira Belo Filho (membro), para examinar o Trabalho de Curso intitulado "ANÁLISE DE ALGORITMO EVOLUTIVO PARA O PROBLEMA DE EMPARELHAMENTO DE CARDINALIDADE MÁXIMA" do estudante ATHOS JOSÉ DE ARAÚJO, Matrícula nº 2016102192010277 do Curso de Bacharel em Ciência da Computação do IF Goiano – Campus Rio Verde. A palavra foi concedida ao estudante para a apresentação oral do TC, houve arguição do candidato pelos membros da banca examinadora. Após tal etapa, a banca examinadora decidiu pela APROVAÇÃO do estudante. Ao final da sessão pública de defesa foi lavrada a presente ata que segue assinada pelos membros da Banca Examinadora.

(Assinado Eletronicamente)

André da Cunha Ribeiro

Orientador(a)

(Assinado Eletronicamente)

Dr. Douglas Cedrim Oliveira

Membro

(Assinado Eletronicamente)

Dr. Márcio Antônio Ferreira Belo Filho

Membro

Observação:

( ) O(a) estudante não compareceu à defesa do TC.

Documento assinado eletronicamente por:

- **Andre da Cunha Ribeiro**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 04/07/2023 20:39:40.
- **Marcio Antonio Ferreira Belo Filho**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 05/07/2023 15:50:36.
- **Douglas Cedrim Oliveira**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 11/07/2023 09:54:30.

Este documento foi emitido pelo SUAP em 04/07/2023. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifgoiano.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 510455
Código de Autenticação: 20d51e6218

INSTITUTO FEDERAL GOIANO
Campus Rio Verde
Rodovia Sul Goiana, Km 01, Zona Rural, 01, Zona Rural, RIO VERDE / GO, CEP 75901-970
(64) 3624-1000

ATHOS JOSÉ DE ARAÚJO

ANÁLISE DE ALGORITMO EVOLUTIVO PARA O PROBLEMA DE EMPARELHAMENTO DE CARDINALIDADE MÁXIMA

Trabalho de curso DEFENDIDO E APROVADO em 04 de JULHO de 2023, pela Banca Examinadora constituída pelos membros:

Dr. MÁRCIO ANTÔNIO FERREIRA BELO FILHO
Instituto Federal Goiano

Dr. DOUGLAS CEDRIM OLIVEIRA
Instituto Federal Goiano

Dr. ANDRÉ DA CUNHA RIBEIRO
Orientador

Rio Verde, GO

2023

AGRADECIMENTOS

Gostaria de expressar minha profunda gratidão a todas as pessoas que contribuíram para a realização deste trabalho e para o sucesso da minha jornada acadêmica. Sem o apoio, orientação e incentivo de cada um de vocês, este trabalho não teria sido possível.

Primeiramente, quero agradecer ao meu orientador, André, por sua valiosa orientação e paciência que foram fundamentais para moldar este me ajudar a superar os desafios que surgiram ao longo do caminho. Sua dedicação à minha formação acadêmica foi inspiradora.

Agradeço à minha família pelo apoio incondicional ao longo de todos esses anos. A paciência e o amor foram a força motriz que me impulsionou a alcançar este marco.

Não posso deixar de mencionar os meus amigos que formei, que compartilharam conhecimento, ideias e risadas ao longo dessa jornada. Os quais também que nos momentos que mais precisava estavam lá para auxiliar. Nossos momentos deixaram uma marca em minhas memórias.

Uma palavra de agradecimento também aos professores e funcionários da instituição, que proporcionaram um ambiente propício ao aprendizado e crescimento pessoal.

Tem dias que é difícil levantar da cama! Eu sinto que não tô ali de verdade, sabe? Parece que eu sou uma espectadora da minha própria vida. Essa minha busca por sentido é tão sem sentido... mas mesmo assim eu sigo buscando...

...

Eu sempre me pergunto como as pessoas aceitam que a vida é assim. Será que eu já devia nascer sabendo ou na verdade ninguém sabe e tá todo mundo fingindo? Eu não sei, eu não consigo aceitar que eu tenho essa única vida e eu não sei o que fazer com ela. Eu tenho essa sensação que mesmo os dias mais longos passam muito rápido e devagar ao mesmo tempo. Parece que eu não vou ter tempo pra fazer tudo o que eu preciso, mas também é tão cansativo viver...

...

Queria olhar para avida de um jeito mais otimista, não queria ir embora dela sem entender o porque de tudo isso...

(Ananda, **Tá todo mundo mal**, 2022)

RESUMO

ARAÚJO, Athos. **ANÁLISE DE ALGORITMO EVOLUTIVO PARA O PROBLEMA DE EMPARELHAMENTO DE CARDINALIDADE MÁXIMA.** 06, 2023. 26 f. Mografia – (Curso de Bacharel em Ciência da Computação), Instituto Federal Goiano - Campus Rio Verde. Rio Verde, GO.

Neste trabalho, exploramos a aplicação de algoritmos evolutivos, especificamente o método (1+1) EE, para resolver o problema de emparelhamento. Investigamos o desempenho deste método em comparação com o algoritmo de Blossom de Edmonds. Por meio de experimentos e análises detalhadas, avaliamos a eficácia e eficiência dessas abordagens na resolução de problemas de emparelhamento. Os resultados mostram um capacidade relativa desse algoritmo, ponderando que este é um problema polinomial para o qual já existe algoritmos muito eficientes.

Palavras-chave: Algoritmo Evolutivo, Grafos, (1+1) EE, Emparelhamento.

ABSTRACT

ARAÚJO, Athos. JOB TITLE. 06, 2023. 26 f. Trabalho de Conclusão de Curso – Bacharel em Ciência da Computação, Instituto Federal Goiano - Campus Rio Verde. Rio Verde, GO, 06, 2023.

In this work, we explore the application of evolutionary algorithms, specifically the (1+1) EA method, to solve the matching problem. We investigate the performance of this method compared to Edmonds' Blossom algorithm. Through experiments and detailed analysis, we assess the effectiveness and efficiency of these approaches in solving matching problems. The results demonstrate a relative capability of this algorithm, considering that it is a polynomial problem for which highly efficient algorithms already exist.

Keywords: Evolutionary Algorithm, Graph, (1+1) EA, Mathcing Problem.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AE	Algoritmo Evolutivo
EE	Estratégias Evolutivas
PRAS	Polynomial-Time Randomized Approximation Schemes

LISTA DE SÍMBOLOS

Δ	Grau máximo do grafo
C	Ciclo
c	Colisões de arestas
E	Arestas
G	Grafo
M	Emparelhamento
P	Caminho
V	Vértices
$\cup$	União de conjunto
$\cap$	Interseção de conjunto
$\subseteq$	Está contido
$\|A\|$	Cardinalidade de conjunto
$\oplus$	Diferença simétrica

LISTA DE ALGORITMOS

SUMÁRIO

1 INTRODUÇÃO

A computação evolutiva tem sido amplamente explorada como uma abordagem potencial para a resolução de problemas complexos de otimização. Os Algoritmos Evolutivos (AE) são uma classe de algoritmos metaheurísticos baseados em população, que procuram emular os princípios evolutivos da seleção natural. Essa abordagem envolve a combinação de elementos como herança genética, mutação e seleção para melhorar progressivamente uma população de soluções candidatas ao longo de várias gerações.

Embora os Algoritmos Evolutivos tenham sido amplamente estudados e aplicados em diversos domínios, a qualidade dos resultados obtidos por esses algoritmos pode variar consideravelmente dependendo do problema em questão. Embora tenham sido relatados casos de sucesso na resolução de problemas complexos, é importante adotar uma visão neutra e considerar as limitações e desafios enfrentados pela computação evolutiva.

No contexto deste trabalho, aplicamos um Algoritmo Evolutivo (AE) para solucionar o problema de emparelhamento de grafos. O objetivo é analisar os resultados obtidos por um AE nesse problema específico. É importante ressaltar que o problema de emparelhamento é um problema polinomial para o qual existem algoritmos altamente eficientes, cujo tempo de execução é delimitado por uma função polinomial de baixo grau. Portanto, nosso objetivo não é propor uma nova abordagem para o problema, mas sim realizar uma análise simples do desempenho dessa classe de algoritmos nesse contexto.

Dentre a variedade de métodos disponíveis, optamos por utilizar o (1+1)-EE (GABRIEL; DELBEM, 2008), que é uma abordagem mais simplista em comparação com outros AE's. O (1+1)-EE é um processo que inicia a partir de um ponto inicial e, em seguida, gera uma nova solução por meio da operação de mutação. Esse processo é repetido até que um critério de parada pré-determinado seja atingido.

Nas capítulos seguintes, apresentaremos um capítulo de conceitos básicos, onde serão explicados os conceitos abordados nas demais seções do trabalho. Em seguida, teremos um capítulo dedicado aos emparelhamentos, abordando a implementação utilizada como parâmetro de comparação de tempo. Posteriormente, será apresentado um capítulo sobre o AE implementado, no qual discutiremos tanto a implementação quanto as questões relacionadas à complexidade. Por fim, concluiremos com um capítulo de discussão dos resultados obtidos.

2 FUNDAMENTAÇÃO

Este capitulo tem como objetivo dar os conceitos básicos sobre Grafos tais como definições de grafos, caminhos, subgrafos como também definições sobre Algoritmos Genéticos como população, cruzamento, mutação.

Grafos

As definições sobre grafos tem como base o livro Feofiloff, Kohayakawa e Wakabayashi (2011). Porém, para a convenção de normalizar alguns símbolos, foram substituídos seguindo o descrito na lista de símbolos. Isso é feito para uma melhor leitura do texto.

Um grafo é um conjunto $G = (V, E)$, onde V é um conjunto arbitrário de *vértices* e E é um subconjunto de V denominado *arestas*, que consiste em pares de *vértices*. Uma *aresta* é denotada por uv, indicando que u incide em v. Consequentemente, u e v são as extremidades da aresta e são vizinhos um do outro.

Subgrafos

Um subgrafo de um grafo G é qualquer grafo H tal que $V(H) \subseteq V(G)$ e $E(H) \subseteq E(G)$.

Grau

O grau de um vértice v é o número de arestas que incidem em v. O grau máximo de um grafo G é denotado por Δ.

Emparelhamento

Um emparelhamento em um grafo é um subgrafo no qual não existem arestas que compartilham vértices. Neste trabalho, consideraremos o emparelhamento apenas como o conjunto de arestas desse subgrafo. Portanto, um emparelhamento é tratado como um subconjunto M de arestas, no qual cada par de arestas não é adjacente.

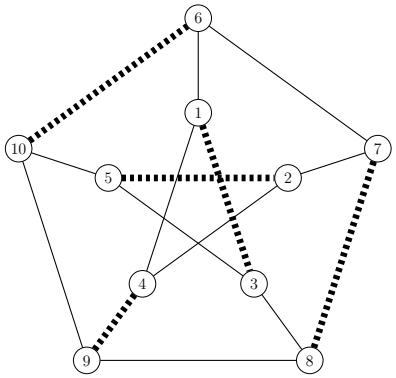


Figura 1 – Grafo de Petersen onde as arestas tracejadas formam um emparelhamento. Autor: Araújo (2023)

Os vértices que estão nos extremos das arestas que pertencem ao emparelhamento são chamados de vértices saturados, enquanto aqueles que não estão nos extremos das

arestas do emparelhamento são chamados de vértices livres. Por exemplo, no emparelhamento formado pelas arestas tracejadas na Figura 1, os vértices $\{1, 2, 3, 4, 5, 6, 7, 9\}$ são saturados, pois estão nos extremos das arestas do emparelhamento, enquanto os vértices $8, 10$ são livres, pois não estão nos extremos das arestas do emparelhamento.

2.0.0.1 Emparelhamento perfeito

Um emparelhamento M em G é perfeito se satura todos os vértices $V(G)$.

2.0.0.2 Emparelhamento maximal

Um emparelhamento M em G é maximal se não for parte de um emparelhamento maior, ou seja, se M em G não for subconjunto próprio de outro emparelhamento M' em G.

Caminhos e ciclos

Um caminho é um subgrafo P definido como $(\{v_1, v_2, v_3, ..., v_n\}, \{v_i v_{i+1} : 1 \leq i \leq n-1\})$. Em outras palavras, um caminho é um grafo P no qual os vértices podem ser rearranjados em uma permutação $\{v_1, v_2, v_3, ..., v_n\}$, de modo que $E(P) = \{v_1v_2, v_2v_3, ..., v_{n-1}v_n\}$. Uma forma de representar um caminho é por meio de uma sequência de vértices, ou seja, um caminho pode ser representado por $P = \{v_1, v_2, v_3, ..., v_n\}$. Na Figura 2, é mostrado um exemplo de caminho com $P = \{1, 2, 3, 4\}$.

Figura 2 – Exemplo de caminho. Autor: Araújo (2023)

Um ciclo é um subgrafo C definido como $(\{v_1, v_2, v_3, ..., v_n\}, \{v_i v_{i+1} : 1 \leq i \leq n-1\} \cup \{v_n v_1\})$, com $n \geq 3$. Em outras palavras, um ciclo é um grafo C com comprimento maior que 3, no qual os vértices podem ser rearranjados em uma permutação $\{v_1, v_2, v_3, ..., v_n\}$, de modo que $E(C) = v_1v_2, v_2v_3, ..., v_{n-1}v_n, v_nv_1$. Na Figura 3, é mostrado um exemplo de ciclo com $C = \{1, 2, 3, 1\}$.

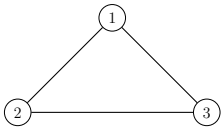


Figura 3 – Exemplo de ciclo. Autor: Araújo (2023)

O comprimento de um caminho ou de um ciclo é o número de arestas do grafo. De forma geral, um caminho de comprimento k possui $k + 1$ vértices, enquanto um ciclo de comprimento k possui exatamente k vértices.

Caminhos alternantes

Dado um grafo G e um emparelhamento M, definimos um caminho alternante como um caminho no qual as arestas se alternam entre as arestas que pertencem a M e as que não pertencem. Em outras palavras, para qualquer par consecutivo de arestas em um caminho alternante, uma delas está em M e a outra não está.

Figura 4 – Exemplo de caminho alternante. Autor: Araújo (2023)

Caminho M-aumentante

Um caminho M-aumentante é um caminho alternante que começa e termina com arestas e vértices livres, o que implica que ele começa e termina com arestas que não estão no emparelhamento. O comprimento de um caminho M-aumentante é sempre ímpar e igual a $2k + 1$, onde k é o número de arestas que pertencem ao emparelhamento M.

No exemplo ilustrado na Figura 5, podemos observar que os vértices 1 e 4 são vértices livres em relação ao emparelhamento considerado. Esses vértices estão nos extremos do caminho M-aumentante representado na figura, o qual começa com a aresta $(1, 2)$ que não está no emparelhamento, percorre uma sequência alternante de arestas e termina com a aresta $(3, 4)$ também fora do emparelhamento.

Figura 5 – Exemplo de caminho de M-aumentante de tamanho 3. Autor: Araújo (2023)

Diferença simétrica

Sejam A e B dois conjuntos. A diferença simétrica entre esses conjuntos é denotada por $A \oplus B$ e é definida como o conjunto composto pelos elementos que estão em A e não estão em B, juntamente com os elementos que estão em B e não estão em A. Matematicamente, podemos expressar a diferença simétrica como $A \oplus B = (A \cup B) - (A \cap B)$. Em outras palavras, a diferença simétrica entre A e B consiste na união dos elementos de ambos os conjuntos, excluindo a interseção dos mesmos.

Uma ilustração visual dessa operação pode ser vista na Figura 6. Nessa figura, os conjuntos A e B são representados e os elementos que estão presentes em apenas um dos conjuntos são destacados. A diferença simétrica $A \oplus B$ é obtida combinando os elementos de A que não estão em B com os elementos de B que não estão em A. A Figura 6 oferece uma representação visual clara desse conceito, auxiliando na compreensão da diferença simétrica entre conjuntos.

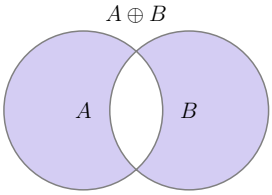


Figura 6 – Diagrama de venn da operação de diferença simétrica. Autor: Araújo (2023)

Existem resultados interessantes aos quais se referem à diferença simétrica em emparelhamentos e caminhos. Dentre os resultados, um nos é muito relevante, que é o Teorema de Berge (1957). Para a prova desse teorema, são necessários dois lemas definidos no livro West (2018). Os resultados são:

Lema 1 *(WEST, 2018) Sejam M um emparelhamento e P um caminho M-aumentante, então $M \oplus P$ é um emparelhamento de cardinalidade $|M| + 1$. Observamos que $P \oplus M$ satura todos os vértices saturados por M.*

Prova. Por definição de caminhos M-aumentante temos que a cardinalidade de P seja $2k+1$. Logo:
$M \oplus P = |(M \cup P)| - |(P \cap M)|$
$M \oplus P = |P| + |M| - |(P \cap M)| - |(P \cap M)|$
$M \oplus P = |P| + |M| - 2|(P \cap M)|$
$M \oplus P = 2k + 1 + |M| - 2k$
$M \oplus P = |M| + 1$ □

Lema 2 *(WEST, 2018) Todo subgrafo de uma diferença simétrica entre dois emparelhamentos é um caminho ou ciclo de comprimento par.*

Prova. Seja M e M', emparelhamentos em um grafo G, e seja $F = M \oplus M'$. Sendo M e M' emparelhamentos, todo vértice terá no máximo uma aresta incidente de cada emparelhamento, logo F terá no máximo duas arestas em cada vértice. Uma vez que $\Delta(F) \leq 2$, todo subgrafo F é um caminho ou ciclo. □

Teorema 3 *(BERGE, 1957) Um emparelhamento M em um grafo G é máximo se, e somente se, não existir um caminho M-aumentante no grafo G.*

Prova. Se M é máximo e G possuí um caminho M-aumentante. Pelo o lema 1, onde temos que $M \oplus P = |M + 1|$, logo isso é um contradição.

Se M não é máximo em um grafo G, então existe um caminho M-aumentante no grafo G. Seja M' um emparelhamento maior que M, ou seja, $|M'| > |M|$. Pelo o lema 2 temos que a diferença simétrica entre esses emparelhamentos é um caminho ou ciclo, como M' é maior que M o subgrafo construído tera mais arestas de M' do que de M, assim, esse subgrafo terá um caminho que inicia com uma aresta de M', alterna com arestas em M e termina com aresta em M', o que é um caminho M-aumentante em G. □

M-Floração

Uma flor é um tipo especial de caminho alternante, representado por $\{v_0, v_1, ..., v_t\}$, onde v_0 é um vértice livre, t é um número ímpar e v_t é igual a um vértice anterior v_i, onde $i < t$. Essa estrutura consiste em uma sequência de vértices $\{v_0, ..., v_i\}$, conhecida como haste, que tem um comprimento par, seguida por um ciclo de vértices ímpar $\{v_i, v_{i+1}, ..., v_t\}$, onde $v_t = v_i$. O vértice v_i é chamado de base da flor. É importante observar que o vértice v_i estará saturado se, e somente se, $i > 0$.

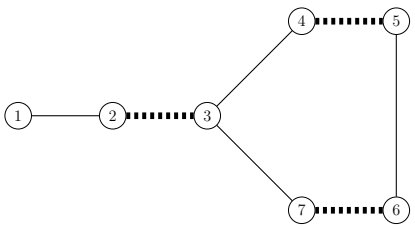


Figura 7 – Grafo de que representa um M-Floração, onde as arestas tracejadas formam um emparelhamento. Autor: West (2018)

Como exemplo ilustrativo, considere a Figura 7. Nessa figura, a flor abrange todo o grafo, enquanto a base da flor é representada pelo caminho $\{1, 2, 3\}$. Por sua vez, a floração é identificada pelo ciclo de comprimento 5, formado pelos vértices $\{3, 4, 5, 6, 7\}$.

Algoritmo Evolutivo

O Algoritmo Evolutivo (AE) é uma classe de algoritmos de otimização que utiliza um mecanismo de busca probabilístico de soluções inspirado no processo de evolução biológica. Embora envolva elementos aleatórios, os AEs se diferenciam dos métodos de busca puramente aleatórios.

Conforme mencionado por Droste, Jansen e Wegener (2002), os AEs são frequentemente empregados para otimizar problemas com funções objetivo estáticas. Existem vários tipos de algoritmos evolutivos, como Algoritmo Genético, Programação Genética e Programação Evolutiva. Como destacado por Bäck e Schwefel (1996), cada tipo de AE representa um paradigma com diferentes definições no espaço de busca.

O conceito fundamental por trás dos AEs é imitar o processo de evolução observado na natureza. Acredita-se que a repetição de etapas de recombinação, mutação e seleção conduz à geração de indivíduos cada vez mais adaptados ao seu ambiente. Nesse contexto, em um AE, uma possível solução para a tarefa de otimização é chamada de indivíduo, e um conjunto desses indivíduos é denominado população. Em cada etapa ou geração do AE, um subconjunto de pais é selecionado com base em suas avaliações na função objetivo, ou seja, em sua aptidão.

Mutação

O operador de mutação desempenha um papel fundamental nos algoritmos evolutivos, sendo responsável por introduzir variações na população e explorar o espaço de busca de soluções. De acordo com Bäck, Fogel e Michalewicz (2018), a mutação age criando novas soluções a partir de pequenas mudanças aleatórias nas soluções anteriores. A mutação é um processo totalmente aleatório e tem como objetivo manter um nível adequado de diversidade na população, garantindo que soluções que possam eventualmente desaparecer tenham a possibilidade de reaparecer (GASPAR-CUNHA; TAKAHASHI; ANTUNES, 2013).

A mutação pode ser aplicada em diferentes níveis de granularidade, desde a mutação de bits individuais até a mutação de estruturas inteiras. A taxa de mutação é um parâmetro crítico que deve ser definido no algoritmo e escolhido cuidadosamente.

Avaliação

Para Gaspar-Cunha, Takahashi e Antunes (2013) e Bäck e Schwefel (1996), a função de avaliação desempenha o papel crucial de avaliar a qualidade de uma solução. Ela determina a sua adequação em relação ao problema em questão. Essa avaliação é realizada de maneira quantitativa, permitindo uma comparação objetiva do mérito relativo entre diferentes soluções. Essa abordagem quantitativa proporciona uma base sólida para a análise e seleção de soluções ao longo do processo evolutivo.

Reparação

A possibilidade de reparar uma solução inválida é uma abordagem de interesse, pois permite manter apenas indivíduos válidos na população. No entanto, essa abordagem requer o desenvolvimento de um algoritmo de reparação específico para o problema em otimização, que seja simples e rápido. Um desafio significativo associado a essa estratégia é a dificuldade potencial em encontrar um algoritmo de reparação adequado.

Colisão

Quando um vértice está saturado por duas ou mais arestas em um emparelhamento, esse emparelhamento é invalido. Aqui usaremos o termo colisão, denotado pelo símbolo c, para descrever um vértice que estará saturado por duas ou mais arestas em um emparelhamento.

3 EMPARELHAMENTO GERAL

Princípios Básicos

O problema de emparelhamento é um problema polinomial que possui uma variedade de algoritmos eficientes para sua resolução. Entretanto, a primeira solução para o problema foi desenvolvida com base no Teorema de Berge (Teorema 3). O Método de Berge (Algoritmo 1) é empregado como uma estrutura fundamental na maioria dos algoritmos.

Algoritmo 1: Método de Berge

Entrada: Grafo G
Saída: Emparelhamento M

```
1 início
2 |  Seja M um emparelhamento em G
3 |  enquanto Existir um caminho M-aumentante P em relação a M faça
4 |  |  M' ← M ⊕ P
5 |  |  M ← M'
6 |  fim
7 fim
```

A ideia, conforme descrita no teorema, é encontrar repetidamente um caminho de M-aumento até que não existam mais caminhos desse tipo. Uma vez que um caminho de M-aumento é identificado, é possível aumentar o emparelhamento em 1.

Atualmente, existem soluções destacadas para abordar esse problema, sendo o algoritmo Hopcroft-Karp (HOPCROFT; KARP, 1973) aplicado a grafos bipartidos, enquanto o algoritmo de Micali-Vazirani (MICALI; VAZIRANI, 1980) é utilizado para grafos gerais.

Implementação

O Algoritmo 2 é uma implementação eficiente do algoritmo Blossom de Edmonds, com base no livro "Teoria Computacional De Grafos" de J.L. Szwarcfiter (SZWARCFITER, 2018). Ele é projetado para encontrar um emparelhamento de cardinalidade máxima em grafos gerais. O procedimento para encontrar esse emparelhamento envolve a verificação da existência de um caminho M-aumentante P em um grafo G dado, onde M é um emparelhamento pré-existente em G.

O Algoritmo 3 chama o procedimento *AUMENTANTE*, que recebe o grafo G e o emparelhamento M como entrada e retorna um caminho P e um conjunto de flores F.

O procedimento "Redução Flor" (Algoritmo 3) desempenha um papel crucial no algoritmo geral de emparelhamento. Ele recebe como entrada um grafo G e um emparelhamento M e retorna um caminho M-aumentante P.

Inicialmente, o procedimento chama o procedimento "Aumentante" (Algoritmo 4) para encontrar um caminho M-aumentante P e um conjunto de vértices F que formam uma flor em relação a M. Essa flor é uma estrutura que contém um ciclo par e permite a redução do grafo.

Em seguida, verifica-se se tanto P quanto F são não vazios. Essa condição indica que há uma flor a ser reduzida. Se essa condição for verdadeira, o grafo original G é

Algoritmo 2: Emparelhamento Geral

```
Entrada: Grafo G
Saída: Emparelhamento M
1 início
2 |  M ← ∅
3 |  repita
4 |  |  P ← ReducaoFlor(G, M)
5 |  |  M ← M ⊕ P
6 |  até P ≠ ∅;
7 fim
```

Algoritmo 3: Redução Flor

```
Entrada: Grafo G, Emparelhamento M
Saída: Caminho P
1 início
2 |  (P, F) ← AUMENTANTE(G, M)
3 |  se (|P| > 0) ∧ (F ≠ ∅) então
4 |  |  G' ← G/F
5 |  |  M' ← M/F
6 |  |  P' ← ReducaoFlor(G', M')
7 |  |  P ← CONSTRUCAO_1(G, G', F, P')
8 |  fim
9 fim
```

reduzido removendo os vértices e as arestas correspondentes ao conjunto F. Isso é feito por meio da operação "$G' = G/F$", onde G' representa o grafo resultante da redução.

Além disso, o emparelhamento M também é atualizado, removendo-se as arestas que fazem parte de F. O novo emparelhamento é armazenado em M'.

Após a redução do grafo e a atualização do emparelhamento, o procedimento "Redução Flor" é chamado recursivamente, agora utilizando o grafo reduzido G' e o novo emparelhamento M'. Essa chamada recursiva busca por mais caminhos M-aumentantes no grafo reduzido, a fim de continuar a busca pelo emparelhamento máximo.

O resultado retornado por essa chamada recursiva é atribuído a P', que representa um caminho M-aumentante no grafo reduzido G'.

Por fim, é realizado o processo "Construção 1" para construir um caminho M-aumentante P no grafo original G. Esse processo envolve a expansão de vértices que foram removidos na redução e a construção de um caminho M-alternante entre esses vértices. O caminho M-aumentante resultante é retornado como o resultado do procedimento "Redução Flor".

O procedimento "Aumentante" (Algoritmos 4) é responsável por construir um caminho M-aumentante e identificar um conjunto de vértices F que formam uma flor em relação ao emparelhamento M.

No início do procedimento, é criado um grafo direcionado auxiliar D_f a partir do emparelhamento M. Esse grafo direcionado é utilizado para auxiliar na construção do caminho M-aumentante.

Em seguida, o procedimento "Construção 3" é chamado, passando como parâmetros o caminho P e o grafo direcionado auxiliar D_f. O objetivo desse processo é construir um

Algoritmo 4: Aumentante

Entrada: Grafo G, Emparelhamento M
Saída: (Caminho P, Ciclo F)
início
  $D_f(M) \leftarrow$ grafo direcionado auxiliar
  $(P, F) \leftarrow CONSTRUCAO\_3(P, D)$
fim

caminho M-aumentante P e identificar um conjunto de vértices F que formam uma flor.

Durante a execução do "Construção 3", é realizado um procedimento de busca a partir do vértice inicial v_0 até o vértice final v_s no grafo direcionado auxiliar D_f. Essa busca é feita de forma que as arestas percorridas sejam do emparelhamento M e que exista uma aresta (v_s, v_t) no grafo original G.

Ao encontrar um caminho P_f que atenda a essas condições, o procedimento "Construção 3"retorna esse caminho como o caminho M-aumentante P. Além disso, o conjunto de vértices F, que representa a flor em relação ao emparelhamento M, também é identificado durante a construção do caminho.

Analise da complexidade

A análise de complexidade do Algoritmo 2, também conhecido como algoritmo de emparelhamento com "Blossom de Edmonds", pode ser realizada da seguinte maneira: Primeiramente, vamos examinar as etapas-chave do algoritmo e determinar o número de passos necessários para cada uma delas.

No procedimento "Aumentante" (Algoritmo 4), a construção do grafo direcionado auxiliar D_f pode ser realizada em $O(m)$ passos, onde m representa o número de arestas no grafo original G. A busca do caminho P_f no grafo direcionado auxiliar e sua transformação em um caminho M-aumentante minimal de G também requerem $O(m)$ passos. Portanto, podemos concluir que o procedimento "Aumentante" termina em $O(m)$ passos.

No procedimento principal, o procedimento "Redução flor" (Algoritmo 3) é chamado recursivamente a partir do algoritmo principal. Cada chamada do procedimento "Redução flor" termina em $O(n(n + m))$ passos, levando em consideração as chamadas recursivas que ele gera. Essa complexidade é justificada pelo fato de que cada redução diminui o emparelhamento em pelo menos dois vértices, e no máximo $O(n)$ chamadas do procedimento são realizadas.

Portanto, a complexidade final do Algoritmo 2 é determinada pela soma dos passos das chamadas do procedimento "Redução flor". Considerando que o número de vértices no grafo original G é n, e o número de arestas é m, podemos concluir que a complexidade final do algoritmo é $O(n^2m)$. Isso significa que o número de passos necessários para executar o algoritmo está relacionado de forma polinomial com o tamanho do grafo, tornando-o um algoritmo eficiente para encontrar um emparelhamento de cardinalidade máxima em grafos gerais.

4 (1+1) EE APLICADO A EMPARELHAMENTO

O objetivo do presente modelo é investigar a convergência do problema de emparelhamento utilizando o (1+1) EE. É importante ressaltar que esse é um problema polinomial para o qual já existem algoritmos muito eficientes, cujo tempo de execução é limitado por uma função polinomial de baixo grau. Portanto, a aplicação proposta do AE pode parecer redundante.

No entanto, este trabalho tem um caráter investigativo, no qual não há intenção de propor um novo método que substitua os métodos existentes, mas sim de analisar o comportamento desse método em relação a esse problema específico. O objetivo principal é compreender as características e limitações do (1+1) EE em um contexto de emparelhamento.

Princípios Básicos

O algoritmo (1+1) EE é conhecido por sua simplicidade, eficiência e facilidade de implementação. No entanto, sua eficácia pode ser restrita em problemas com múltiplos ótimos locais, uma vez que o algoritmo pode ficar preso em um ótimo local. O Algoritmo 5 apresenta uma descrição direta da implementação do problema neste trabalho. Nos parágrafos a seguir, forneceremos uma descrição mais detalhada de cada etapa do algoritmo.

Algoritmo 5: (1+1) EE

```
1 p_n ← 1/n;
2 Escolha aleatoriamente x ∈ {0,1}^n;
3 repita
4 |   Calcule x' ← trocando independentemente cada bit x_i com uma
  |     probabilidade p_n;
5 |   se f(x') > f(x) então
6 |   |   x ← x';
7 |   fim
8 até Condição;
```

Primeiramente, o algoritmo inicia com a geração de uma solução inicial. Essa solução pode ser gerada aleatoriamente ou por meio de uma heurística específica, dependendo da natureza do problema em questão. Em seguida, é avaliada a qualidade da solução inicial utilizando uma função objetivo, a qual deve ser definida de acordo com os objetivos do problema.

Após a obtenção da solução inicial, o algoritmo entra em um *loop* principal, no qual são realizadas gerações para melhorar a solução atual. Nessa etapa, é gerada uma nova solução por meio de uma mutação aplicada à solução atual. Essa mutação pode envolver alterações aleatórias ou heurísticas específicas, dependendo das características do problema.

Após a geração da nova solução, é realizada uma avaliação da qualidade dessa solução utilizando a função objetivo. Se a nova solução for melhor que a solução atual, ela substitui a solução atual. Caso contrário, a solução atual é mantida. Essa estratégia de aceitar apenas soluções melhores, conhecida como "melhoramento", é fundamental para o algoritmo buscar uma solução ótima.

O algoritmo continua gerando novas soluções e avaliando a qualidade delas ao longo de várias gerações, até que um critério de parada seja alcançado. Esse critério de parada pode ser um número máximo de gerações, um limite de tempo ou uma condição específica definida pelo problema em análise. Ao atingir o critério de parada, o algoritmo retorna a melhor solução encontrada durante as gerações.

Implementação

Algoritmo 6: (1+1) EE para emparelhamento

```
1  p_m ← 1/m;
2  Escolha aleatoriamente x ∈ {0,1}^m, tal que x não possua vértices saturados;
3  i ← 0;
4  repita
5  |  Calcule x' ← trocando independentemente cada bit x_i com uma
   |    probabilidade p_m;
6  |  se f(x') > f(x) então
7  |  |  x ← x';
8  |  fim
9  |  i ← i + 1;
10 até i = 20n;
```

A implementação do algoritmo (1+1) EE baseia-se no uso de um vetor binário que representa a solução, onde cada posição corresponde a uma aresta no grafo. Quando uma posição possui o valor "1", isso indica que a aresta está incluída no emparelhamento.

Antes de realizar os testes principais, conduzimos uma série de experimentos com o objetivo de avaliar o desempenho do algoritmo em diferentes configurações. Durante esses testes, observamos que uma solução inicial gerada de forma completamente aleatória resultava em um emparelhamento inválido, com várias colisões, ou seja, duas arestas compartilhando o mesmo vértice. A resolução dessas colisões mostrou-se uma etapa extremamente custosa.

A fim de evitar essas colisões, fizemos a operação semelhante a de reparação. Optamos por selecionar apenas arestas que não fazem parte do emparelhamento inicial. Utilizando uma lista indexada, somos capazes de verificar em tempo constante (complexidade $O(1)$) se um vértice de uma aresta selecionada já está saturado.

Algoritmo 7: Inicialização

```
   Entrada: Número de arestas m
   Saída: Emparelhamento M
1  início
2  |  para i ← 0 até m faça
3  |  |  se M_i então
4  |  |  |  M_i ← Verdadeiro;
5  |  |  fim
6  |  fim
7  fim
```

A operação de mutação do algoritmo foi mantida como descrita anteriormente. Para cada aresta no emparelhamento, há uma probabilidade p_m de selecionar ou remover

essa aresta. Nessa etapa do processo, não consideramos se algum dos vértices da aresta selecionada está saturado.

Algoritmo 8: Mutação

Entrada: Emparelhamento M, número de arestas m, probabilidade de mutação p_m
Saída: Emparelhamento M

```
1 início
2     para i ← 0 até m faça
3         p ← random(0, 1);
4         se p < p_m então
5             M_i ← ¬M_i
6         fim
7     fim
8 fim
```

A avaliação da solução é determinada pela cardinalidade do emparelhamento, que é calculada somando-se todas as variáveis x_i no vetor de solução, conforme a expressão $\sum_{i=0}^{m} x_i$. No entanto, devido ao processo aleatório e à tentativa de explorar uma região mais ampla em busca de uma solução, podem ocorrer colisões. Nesse caso, retornamos um número negativo para indicar a presença de colisões na solução.

Algoritmo 9: Avaliação

Entrada: Emparelhamento M, número de arestas m
Saída: Cardinalidade do emparelhamento S

```
1  início
2      F ← 0;
3      C ← 0;
4      para i ← 0 até m faça
5          se M_i = Verdadeiro então
6              F ← S + 1
7          senão
8              C ← C + 1
9          fim
10     fim
11     se C > 0 então
12         S ← −C
13     senão
14         S ← F
15     fim
16 fim
```

A condição de parada do *loop* de gerações é definida como $20n$. Essa escolha foi baseada em experimentos anteriores nos quais observamos que a solução começa a convergir para um ótimo local após $10n$ iterações.

Analise da complexidade

A analise da complexidade do Algoritmo 6, que é o (1+1) EE aplicado a emparelhamento, pode ser realizada da seguinte maneira. Primeiro, temos que observar que há

um custo para gerar uma solução inicial, esse custo está no orem de $O(m)$ onde m é a cardinalidade das arestas do grafo. Nesse método há à verificação de quais vértices já estão emparelhados conforme é criada a solução inicial. Porém, com o uso de uma lista indexada é possível manter essa verificação na ordem de $O(1)$, o que nos mantém a linearidade no processo de criação da solução inicial, com uma complexidade de $O(m)$.

Após o processo de criação da solução inicial entramos em um *loop* que nos dá as gerações do nosso algoritmo, assim como já evidenciado, o critério de parada utilizado foi o limite de $20n$ gerações. Assim, temos que mesmo com uma constante alta, o *loop* por si só tem uma complexidade de $O(n)$.

Aninhado ao *loop* de gerações temos o procedimento de "mutação" (Algoritmo 8). Assim como a geração de uma solução inicial, ele também caminha por essa solução fazendo o processo de *flip*, de tal forma que esse processo também seja é executado de forma linear na ordem de $O(m)$.

O procedimento de avaliação (Algoritmo 9), assim como o de mutação ela é aninhada ao *loop* de gerações. Ela caminha por toda a solução somando os valores de cada posição, porém também é feita uma avaliação se há uma colisão, mas como já mostrado esse processo de verificar colisão pode ser feito de ordem $O(1)$, assim temos que o procedimento também é linear e tem uma complexidade de $O(m)$.

Ao analisar os procedimentos presentes, podemos determinar que a complexidade assintótica total é de $O(m + n(m + m))$. Simplificando essa expressão, chegamos à complexidade $O(nm)$ para todo o processo.

5 RESULTADOS E DISCUSSÕES

Os algoritmos foram implementados na linguagem de programação Python. Os experimentos foram realizados em um computador com o sistema operacional *Linux Mint 19.1* e processador *Intel Core i5-3470*. Os objetivos desses experimentos foram avaliar a eficácia do algoritmo $(1 + 1)AE$ na resolução do problema de emparelhamento de cardinalidade máxima. Para a organização desse trabalho, as imagens referentes as curvas de convergência esta localizada no Apêndice A.

Para fins de comparação, foi implementado o algoritmo "Emparelhamento em grafos gerais"descrito por Szwarcfiter (2018), cuja complexidade é de ordem $O(n^2m)$.

É importante ressaltar que os resultados deste trabalho corroboram com os resultados apresentados no trabalho de Giel e Wegener (2003). Ambos utilizam a mesma técnica para resolver o problema, porém existe uma diferença significativa na solução inicial. Enquanto este trabalho parte de uma solução válida, o trabalho deGiel e Wegener (2003) parte de uma solução aleatória obtida por uma distribuição uniforme. Além disso, Giel e Wegener (2003) menciona que são necessárias $\sqrt{c}$ iterações para converter colisões em um emparelhamento válido. Como já observado, a solução aleatória gera um grande número de colisões, mesmo com um alto número de gerações, o que inviabilizou o seu uso neste trabalho.

Outro ponto a ser destacado é que o trabalho do Giel e Wegener (2003) considera o (1+1) AE como um método de aproximação aleatória em tempo polinomial (*Polynomial-Time Randomized Approximation Schemes* - PRAS), que encontra um emparelhamento em tempo $O(m^2)$. No entanto, este trabalho não corrobora essa afirmação, uma vez que a melhor solução obtida, conforme mostrado na Tabela 2, foi de 96.6%, ou seja, de um emparelhamento de cardinalidade 100, este trabalho encontraria um emparelhamento de cardinalidade próxima a 96. É importante salientar que, mesmo em testes preliminares, quando o critério de parada do algoritmo era o emparelhamento máximo, o mesmo executou uma mesma instância por mais de 3 horas, após esse tempo optamos por não continuar com o teste.

Para avaliar o desempenho dos algoritmos, foram realizados experimentos em seis grafos diferentes, sendo três esparsos e três densos. Os grafos possuíam 500, 1000 e 1500 vértices, respectivamente. Estes grafos foram gerados de forma aleatória partir de uma distribuição uniforme. É garantido que foi garantido que todos os grafos gerados são conexo com grau mínimo e máximo próximos. É de salienta que todos os grafos gerados dessa forma eram grafos com um emparelhamento perfeito as causas não são tão claras bem como não é o objetivo desse trabalho buscar o motivo para isso. Para cada grafo, o algoritmo foi executado 10 vezes e os resultados foram registrados.

A Tabela 1 apresenta os resultados dos testes realizados com grafos esparsos, nos quais a solução aproximou-se de 90.8% da solução máxima. Os desvios padrão para cada grafo foram de 2.5, 2.4 e 3.9, respectivamente. A Tabela 2 mostra os resultados dos testes realizados com grafos densos, nos quais a solução aproximou-se de 96% da solução máxima. Os desvios padrão para cada grafo foram de 2.4, 1.9 e 4.1, respectivamente.

Considerando os resultados apresentados, é possível perceber que a abordagem utilizada apresenta limitações, uma vez que não foi capaz de encontrar um emparelhamento máximo em nenhum dos testes realizados. No entanto, é importante destacar que esta foi a forma mais simples de implementação do algoritmo, e que existem métodos de mutação que podem ser explorados para ampliar a região de busca e melhorar o desempenho do

Tabela 1 – Resultados com grafos esparsos.

	Algoritmo Evolutivos ($O(nm)$)						Szwarcfiter (2018) ($O(n^2m)$)	
Grafos	25%	50%	75%	100%	aproximação (%)	tempo (s)	tamanho	tempo (s)
500 vértices	227	229	231	233	91.6%	4.6336	250	4.3281
1000 vértices	450	452	453	455	90.2%	19.0474	500	19.2997
1500 vértices	676	679	684	686	90.7%	42.6634	750	47.5928

Tabela 2 – Resultados com grafos densos.

	Algoritmo Evolutivos ($O(nm)$)						Szwarcfiter (2018) ($O(n^2m)$)	
Grafos	25%	50%	75%	100%	aproximação (%)	tempo (s)	tamanho	tempo (s)
500 vértices	238	238	240	243	95.4%	21.2140	250	31.0651
1000 vértices	480	480	482	484	96.1%	281.7204	500	396.3679
1500 vértices	722	723	727	733	96.6%	1300.8959	750	1427.4136

algoritmo.

O trabalho de Giel e Wegener (2003) destaca a abordagem do Simulated Annealing, que explora apenas vizinhos de Hamming, ou seja, soluções que diferem em apenas um elemento. Esse método às vezes precisa aceitar combinações piores durante a busca. Por outro lado, os algoritmos evolutivos frequentemente consideram novos pontos de busca com uma distância de Hamming maior em relação ao ponto de busca atual. Essa diferença na estratégia de busca pode impactar significativamente a eficiência e eficácia do algoritmo em encontrar soluções ótimas ou próximas do ótimo.

Outro ponto relevante a ser considerado é que, embora não seja possível afirmar com certeza se os ótimos locais encontrados pelo algoritmo são emparelhamentos maximais, é possível observar que a busca sempre converge para um ótimo local. As curvas de convergência apresentadas no Apêndice A demonstram que as soluções estabilizam em um ponto ótimo local, mesmo após várias gerações.

No entanto, ao analisar os dados das tabelas (Tabela 1 e Tabela 2), é possível inferir que o algoritmo está sendo eficiente em encontrar soluções de boa qualidade, mesmo que não sejam as melhores possíveis. Esses resultados indicam que o algoritmo está explorando de forma satisfatória o espaço de busca e encontrando emparelhamentos próximos da solução ótima. Isso não tira o fato de ser uma problema com uma solução exata, o que exige que seja feito melhores estudos para usar métodos que ampliem ainda mais a região de busca e melhorar o desempenho do algoritmo.

É importante ressaltar que, embora os ótimos locais encontrados possam não ser maximais, eles ainda podem ser considerados soluções valiosas para o problema em questão. Em muitos casos, obter um emparelhamento de alta qualidade, mesmo que não seja o máximo, pode ser suficiente para atender aos requisitos práticos e fornecer resultados úteis para aplicações reais. Outro ponto é a criação de soluções hibridas, uma vez que, encontrado uma solução em um ótimo local ou a solução convergir para um platô podemos partir dessa solução com um método exato.

Portanto, os resultados das tabelas indicam que o algoritmo implementado está desempenhando de maneira satisfatória para um AE, porém não é satisfatório para o problema em questão. Encontrando soluções que são boas aproximações do ótimo, mesmo que não sejam as soluções máximas absolutas.

Os quartis das execuções fornecem informações sobre a dispersão dos dados e sua tendência central, revelando aspectos importantes da consistência e confiabilidade das soluções encontradas. Além disso, o desvio padrão, uma medida que avalia o grau de variação dos dados em relação à média, complementa essas análises, demonstrando a

estabilidade dos resultados obtidos e a sua capacidade de serem reproduzidos em diferentes contextos.

Ao observar as curvas de convergência das execuções realizadas, notamos uma notável proximidade entre os resultados da pior, melhor e mediana execução. Essa convergência semelhante indica uma robustez significativa na solução proposta. Esse comportamento uniforme das curvas sugere que os resultados obtidos são consistentes e confiáveis, reforçando a confiança na replicabilidade dos resultados em diferentes iterações ou aplicações.

Por fim, a análise da complexidade do algoritmo em relação a uma outra abordagem utilizada para fins de comparação mostra que, mostra que o (1+1) AE apresentado nesse trabalho apresenta uma complexidade menor, o que significa que o tempo de execução tende a ser reduzido à medida que o grafo cresce com o volume de vértices. Entretanto toda via isso não exime do fato da solução encontrada ser uma solução aproximada, além que há abordagens como a do Micali e Vazirani (1980) que apresenta uma complexidade menor em relação ao apresentado e encontra uma solução exata para o problema.

6 CONCLUSÃO

Este trabalho teve como objetivo analisar a aplicação do (1+1) AE no problema de emparelhamento de cardinalidade máxima, utilizando como base de comparação uma solução já existente para o problema.

Uma vez que o problema em questão já possui diversas soluções conhecidas com complexidade de baixo grau, não há justificativa para o uso de um algoritmo que tende a se aproximar da resposta.

Os resultados deste trabalho corroboram com essa ideia, uma vez que o (1+1) AE não foi capaz de encontrar a melhor solução para o problema.

Trabalhos Futuros

Dada a finalidade do algoritmo, o (1+1) Algoritmo Evolutivo (AE) demonstra potencial para gerar soluções de qualidade em problemas NP-Completos. Entre esses problemas, destaca-se o de Conjunto Independente, devido à sua estrutura de dados ser semelhante ao desenvolvido neste trabalho.

Referências

BERGE, C. Two theorems in graphs theory. **Proceedings of the National Academy of Sciences**, v. 43, p. 842–844, 1957. Citado 2 vezes nas páginas 4 e 5.

BäCK, T.; FOGEL, D. B.; MICHALEWICZ, Z. **Evolutionary computation 1: Basic algorithms and operators**. 1. ed. [S.l.]: CRC Press, 2018. Citado na página 6.

BäCK, T.; SCHWEFEL, H.-P. Evolutionary computation: an overview. **Proceedings of IEEE International Conference on Evolutionary Computation**, p. 20–29, 1996. Citado na página 6.

DROSTE, S.; JANSEN, T.; WEGENER, I. On the analysis of the (1+1) evolutionary algorithm. **Theoretical Computer Science**, London, UK, v. 276, p. 51–81, 2002. Citado na página 6.

FEOFILOFF, P.; KOHAYAKAWA, Y.; WAKABAYASHI, Y. **Uma introdução sucinta à teoria dos grafos**. São Paulo: [s.n.], 2011. Citado na página 2.

GABRIEL, P. H. R.; DELBEM, A. C. B. **Fundamentos de algoritmos evolutivos**. São Carlos: ICMC-USP, 2008. Citado na página 1.

GASPAR-CUNHA, A.; TAKAHASHI, R.; ANTUNES, C. **Manual de computação evolutiva e metaheurística**. 1. ed. Belo Horizonte: Editora UFMG, 2013. Citado na página 6.

GIEL, O.; WEGENER, I. Evolutionary algorithms and the maximum matching problem. **Annual Symposium on Theoretical Aspects of Computer Science**, Berlin, Heidelberg, v. 2607, p. 415–426, 2003. Citado 2 vezes nas páginas 15 e 16.

HOPCROFT, J. E.; KARP, R. M. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. **SIAM Journal on Computing**, v. 2, n. 4, p. 225–231, 1973. Citado na página 8.

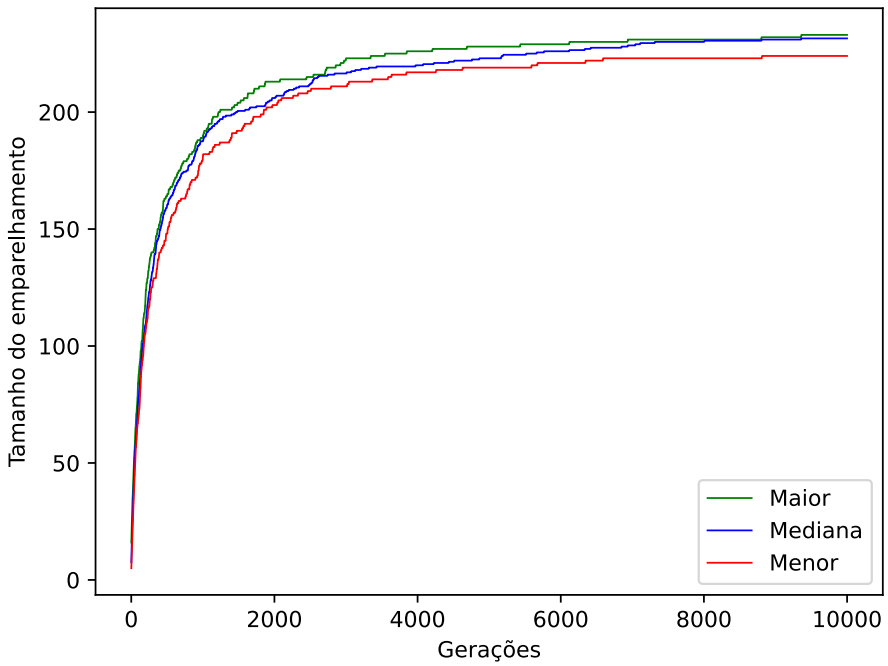
MICALI, S.; VAZIRANI, V. V. An $o(\sqrt{n}m)$ algorithm for finding maximum matching in general graphs. **Annual IEEE Symposium on Foundations of Computer Science**, Syracuse, NY, USA, p. 17–27, 1980. Citado 2 vezes nas páginas 8 e 17.

SZWARCFITER, J. **Teoria computacional de grafos**. 1. ed. Rio de Janeiro: Elsevier, 2018. Citado 3 vezes nas páginas 8, 15 e 16.

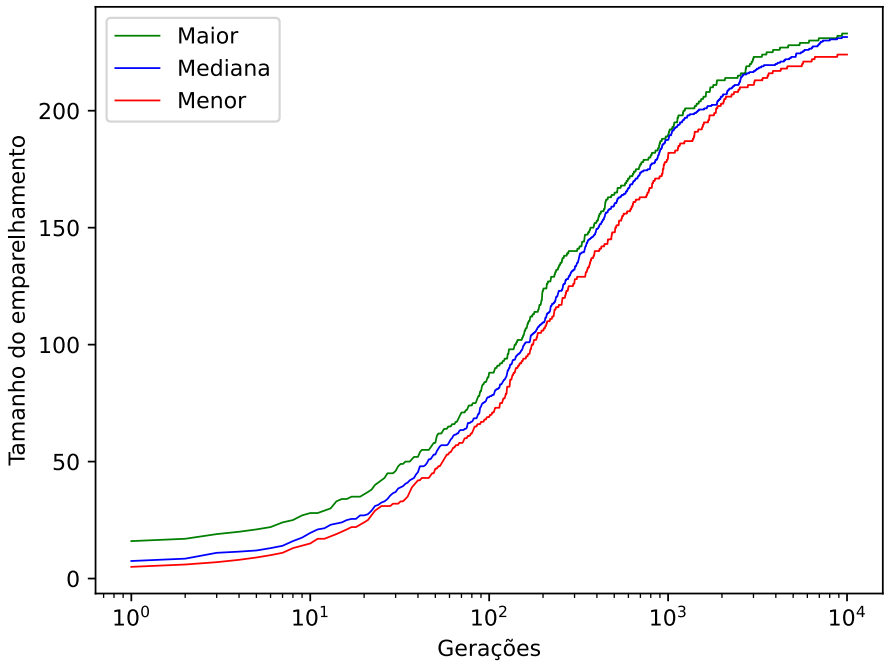
WEST, D. **Introduction to Graph Theory**. 2. ed. [S.l.]: Pearson, 2018. Citado 3 vezes nas páginas , 4 e 5.

Apêndices

APÊNDICE A – Curvas de convergência

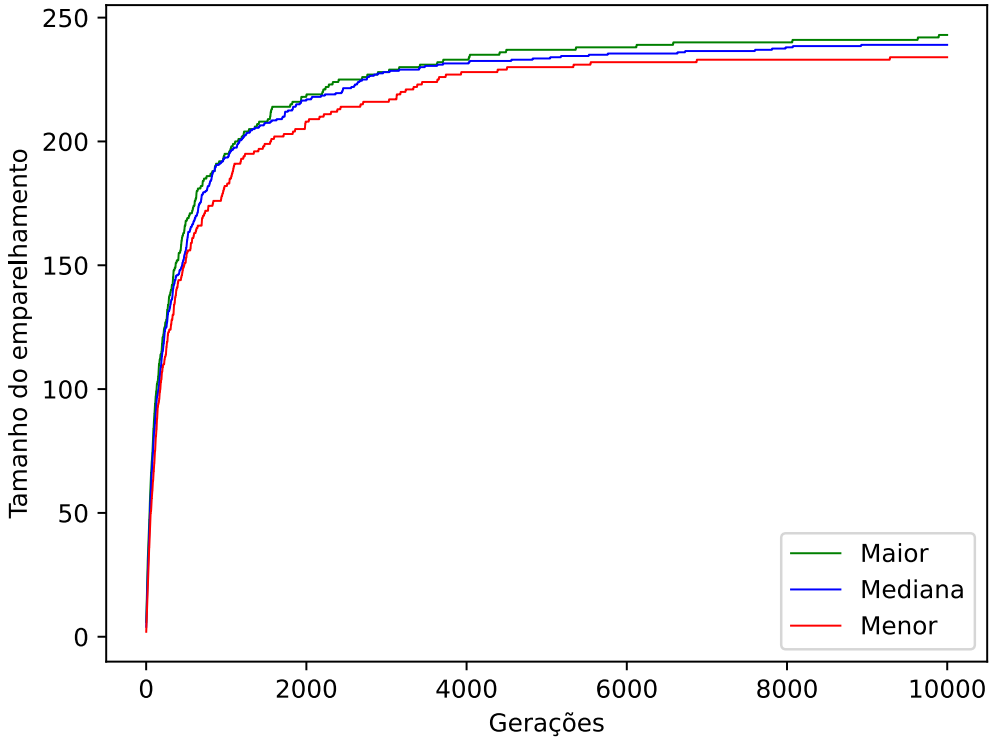


(a) Curva de convergência em escala padrão.

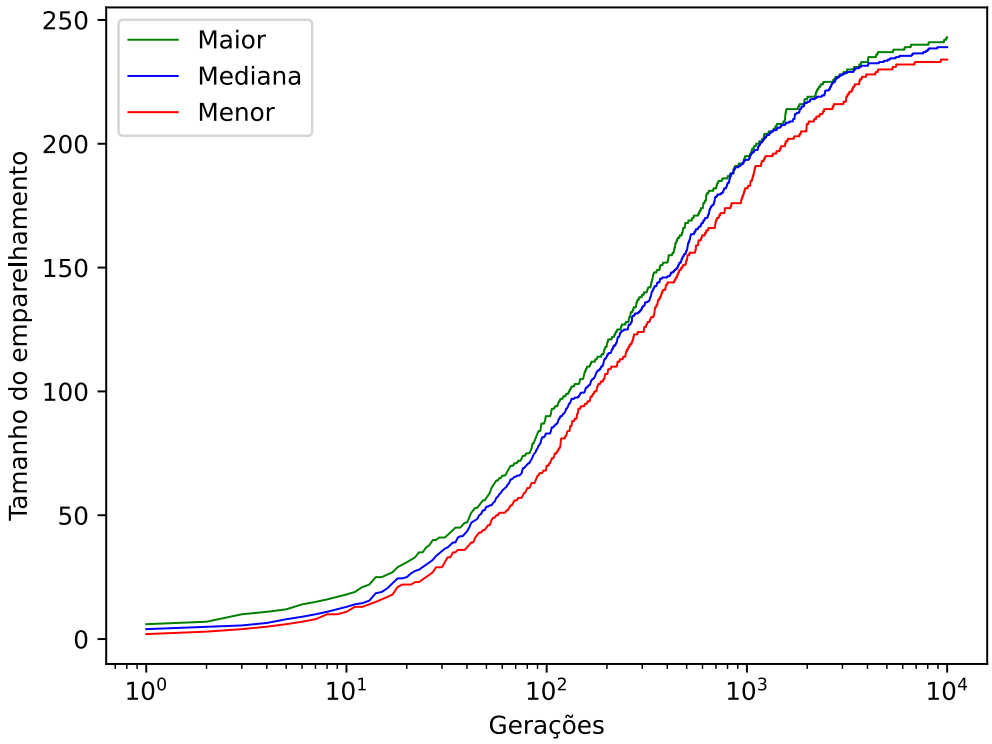


(b) Curva de convergência em escala logarítmica.

Figura 8 – Execução grafo esparso de 500 vértices. Autor: Araújo (2023)

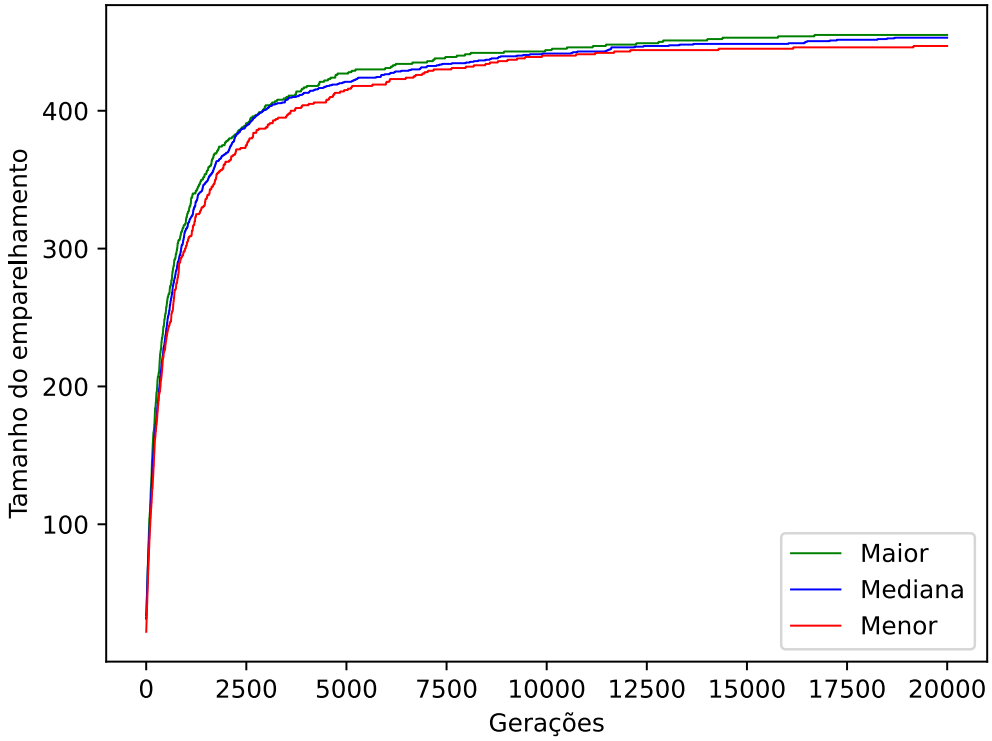


(a) Curva de convergência em escala padrão.

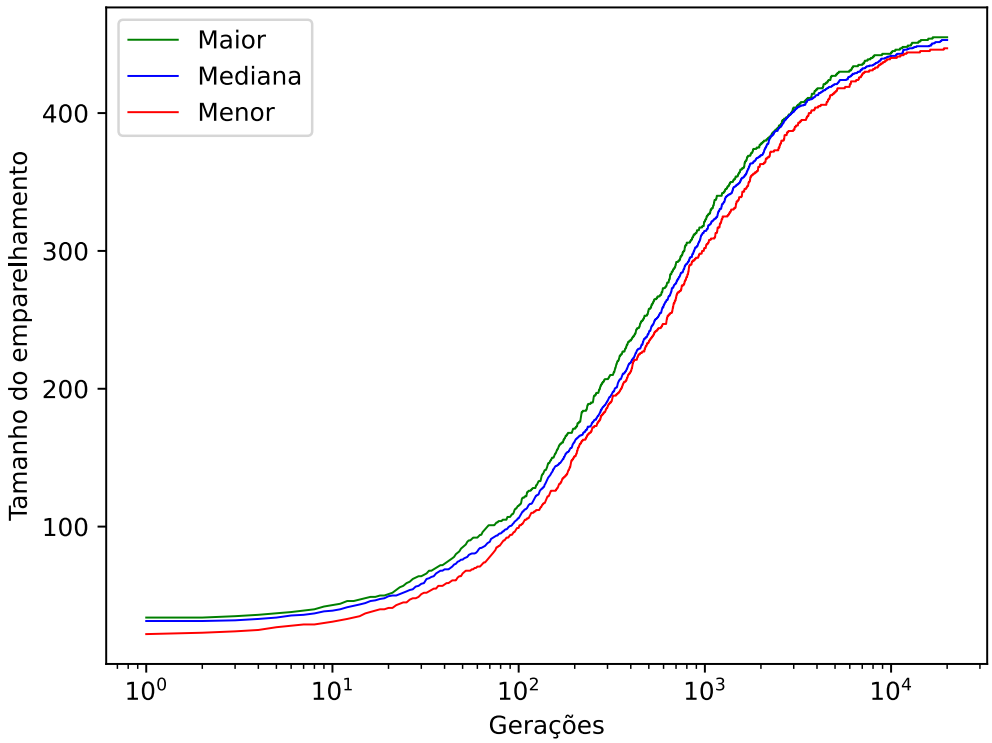


(b) Curva de convergência em escala logarítmica.

Figura 9 – Execução grafo denso de 500 vértices. Autor: Araújo (2023)

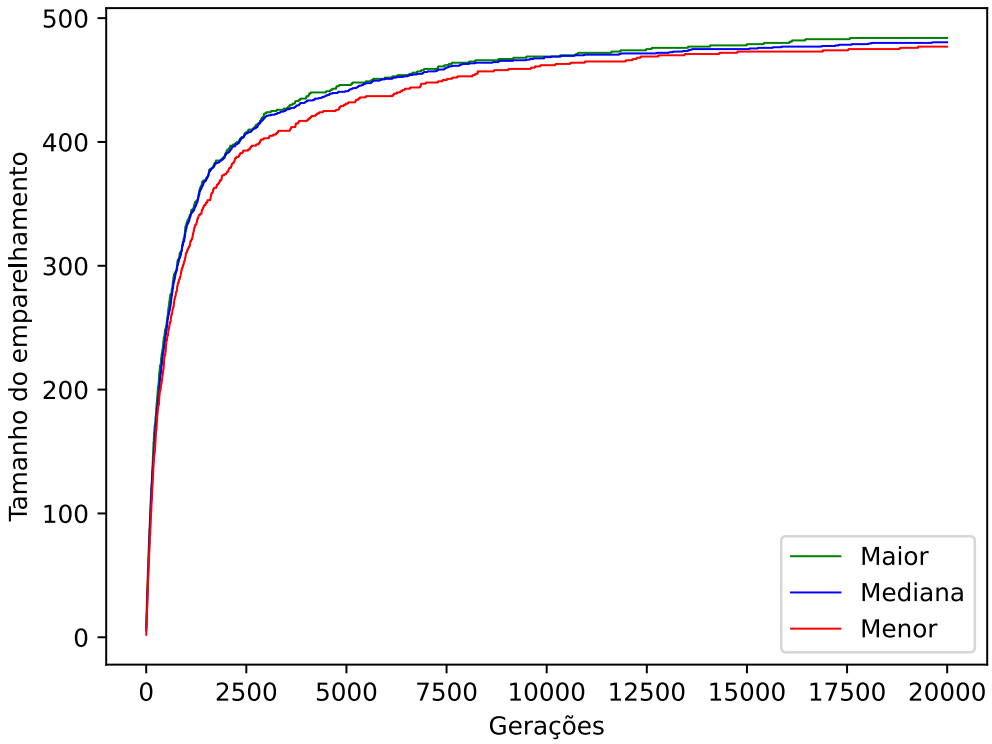


(a) Curva de convergência em escala padrão.

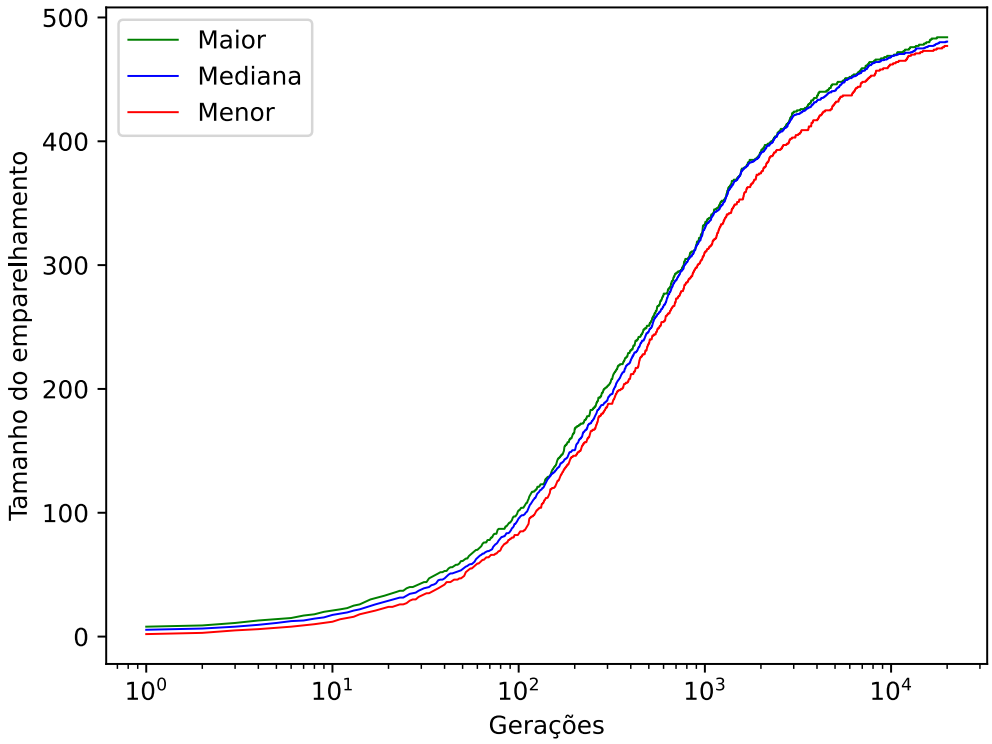


(b) Curva de convergência em escala logarítmica.

Figura 10 – Execução grafo esparso de 1000 vértices. Autor: Araújo (2023)

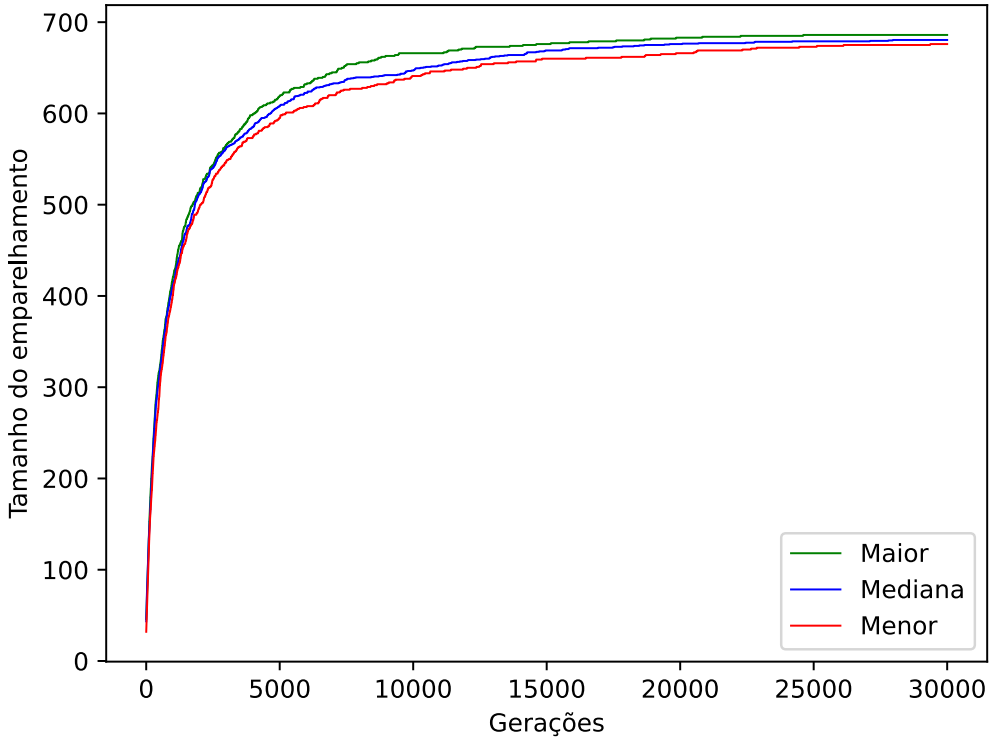


(a) Curva de convergência em escala padrão.

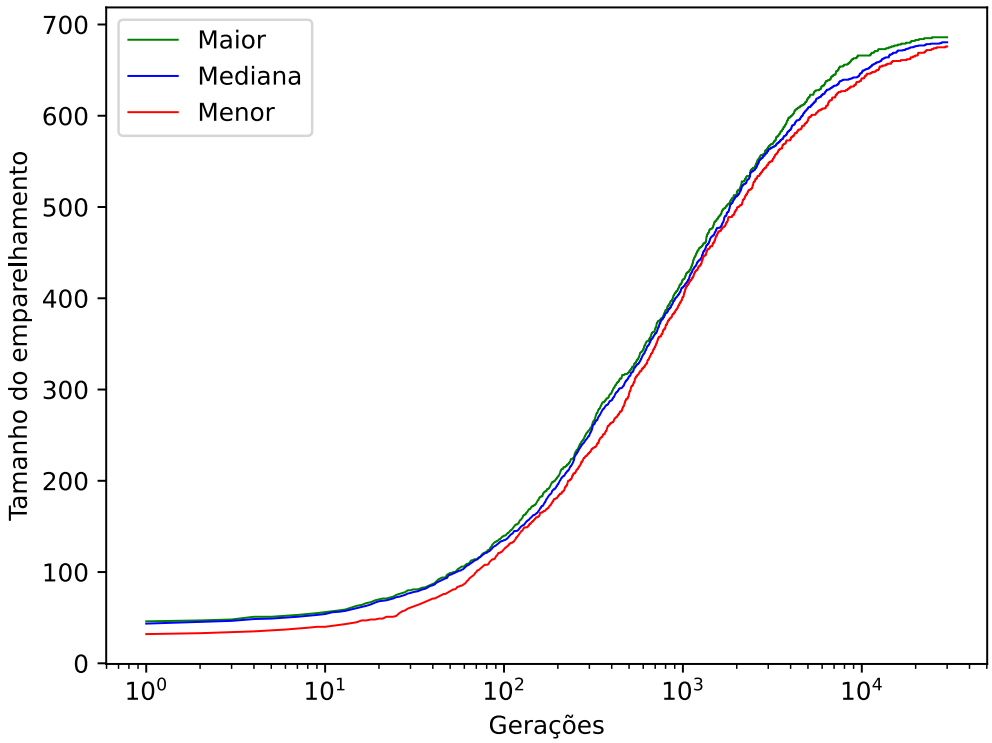


(b) Curva de convergência em escala logarítmica.

Figura 11 – Execução grafo denso de 1000 vértices. Autor: Araújo (2023)

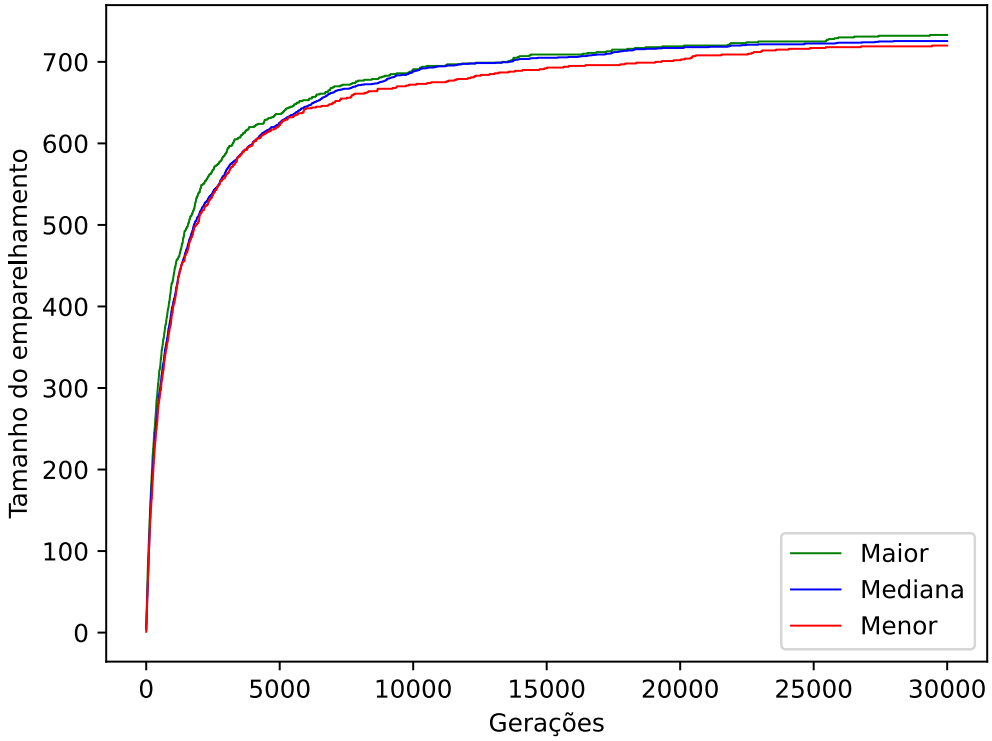


(a) Curva de convergência em escala padrão.

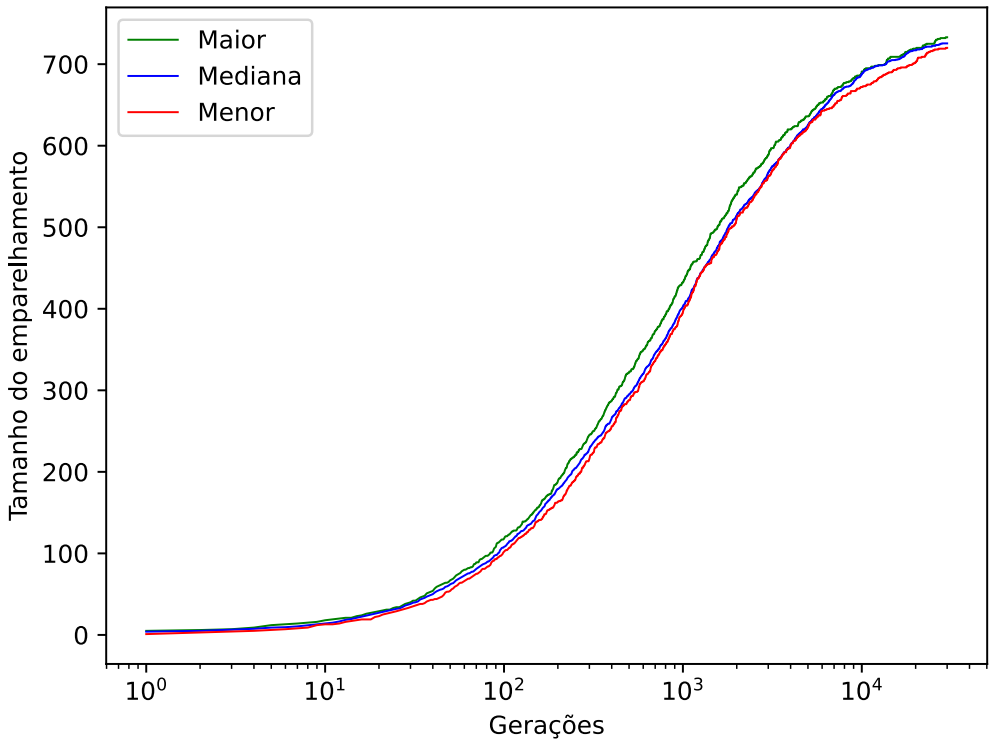


(b) Curva de convergência em escala logarítmica.

Figura 12 – Execução grafo esparso de 1500 vértices. Autor: Araújo (2023)



(a) Curva de convergência em escala padrão.



(b) Curva de convergência em escala logarítmica.

Figura 13 – Execução grafo denso de 1500 vértices. Autor: Araújo (2023)