Relatório 4 - Métodos Heurísticos para Redução de Largura de Banda

Adriano D. Goulart[1]
Graduando em Ciência da Computação
`adriano.goulart@estudante.ufla.br`

Claudinei M. da Silva[1]
Graduando em Ciência da Computação
`claudinei.silva@estudante.ufla.br`

[1]Departamento de Ciência da Computação - Universidade Federal de Lavras (UFLA)

Abstract. *This report addresses heuristics to address the Bandwidth Minimization problem. Using a genetic algorithm as a meta-heuristic to improve the results of the Cuthill-Mckee algorithm.*

Resumo. *Este relatório aborda heurísticas para tratar o problema de Minimização da Largura de Banda. Utilizando um algoritmo genético como meta-heurística para aprimorar os resultados do algoritmo de Cuthill-Mckee.*

1. Introdução

A minimização da largura de banda é um problema que consiste em, rotular os vértices de um gráfico com os inteiros de 0 a n (n é o número de vértices) de modo a minimizar a maior diferença entre rótulos de vértices adjacentes. O problema é NP-Completo, e existe vários algoritmos exatos e heurísticas para resolvê-lo. Este trabalho aborda um algoritmo genético classificado como uma meta-heurística, que trabalha em cima do algoritmo de Cuthill-Mckee para conseguir melhores resultados para a minimização da largura de banda.

2. Reverse Cuthill - McKee

O algoritmo de Cuthill-Mckee é usado para ordenar as incógnitas e equações em sistemas de equações lineares positivas esparsas. Para um determinado sistema linear Ax = b, este algoritmo é projetado para produzir uma matriz de permutação P tal que PAP^T [W. Liu 2006]. Ele é baseado no algoritmo de Busca em Largura de um grafo, cuja matriz de adjacência é a versão esparsa da matriz quadrada de entrada. Com uma renumeração apropriada dos nós, geralmente é possível produzir uma matriz com uma largura de banda muito menor.

A diferença entre o algoritmo Cuthill-Mckee reverso é que os índices finais obtidos usando o algoritmo Cuthill-Mckee são revertidos. Esta simples modificação geralmente resulta em uma ordenação melhor em termos de eficiência, apesar da largura de banda continuar a mesma.[W. Liu 2006]

Abaixo estão as etapas do algoritmo Reverse Cuthill-Mckee reverso baseado em [E. Cuthill 1969] e [RCM]:

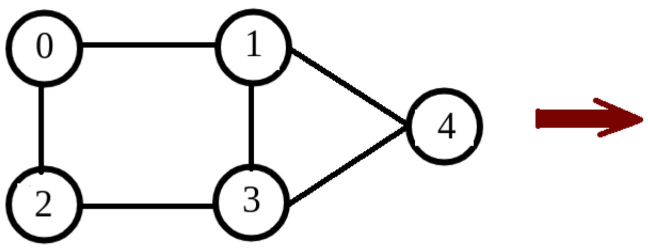
1. Instanciar uma fila vazia **Q** e matriz vazia para a permutação dos objetos **R**
2. Em primeiro lugar, encontrar o objeto com grau mínimo cujo índice ainda não foi adicionado a **R**. Digamos, o objeto correspondente à enésima linha tenha sido identificado como o objeto com um grau mínimo. Adicionar **p** a **R**.
3. Quando um índice é adicionada a **R**, adicionamos todos os vizinhos do objeto correspondente no índice. Os vizinhos são nós com valor diferente de zero entre os elementos não diagonais na enésima linha.
4. Extraia o primeiro nó em **Q**, digamos **C**. Se **C** não foi inserido em **R**, adicione-o a **R**, adicione a **Q** os vizinhos de **C** em ordem crescente de grau.
5. Se **Q** não estiver vazio, repita **linha 4**.
6. Se **Q** estiver vazio, mas houver objetos na matriz que não foram incluídos no **R** , inicie novamente na **linha 2**. (Pode acontecer se houver grafos disjuntos).
7. O algoritmo encerra assim que todos os objetos forem incluídos em **R**.
8. Finalmente, inverta os índices em **R** ,ou seja (***swap*** (R [i], R [P-i + 1])).

Para o exemplo abaixo, o grau de um nó é definido como a soma dos elementos não diagonais na linha correspondente. Então para um grafo qualquer, o grau de um nó **A** é o número de nós conectados a ele.

Dada uma matriz simétrica:

$$\begin{bmatrix} 0,0 & 0,78 & 0,79 & 0,8 & 0,23 \\ 0,9 & 0,0 & 0,43 & 0,77 & 0,75 \\ 0,82 & 0,0 & 0,0 & 0,79 & 0,34 \\ 0,8 & 0,8 & 0,8 & 0,0 & 0,8 \\ 0,54 & 0,97 & 0,12 & 0,78 & 0,0 \end{bmatrix}$$

A *esparsificação* para uma matriz é definida como: se o elemento da matriz em i, j tiver um valor menor que 0,75 vira 0, caso contrário, 1.



$$\begin{array}{c|ccccc|} & 0 & 1 & 2 & 3 & 4 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 2 & 1 & 0 & 0 & 1 & 0 \\ 3 & 1 & 1 & 1 & 0 & 1 \\ 4 & 0 & 1 & 0 & 1 & 0 \end{array}$$

Figura 1. Matriz após Esparsificação

Grau de nó $0 = 2,6$

Grau de nó $1 = 2,803$

Grau de nó $2 = 2,55$

Grau de nó $3 = 3,2$

Grau de nó $4 = 2,41$

Ordem de permutação de objetos (R): $0 \quad 2 \quad 1 \quad 3 \quad 4$

A nova ordem de permutação é apenas a ordem dos nós, ou seja, converte o nó R [i] para o nó i. Portanto, converta o nó R [0] = 0 para 0; nó R [1] = 2, para 1; nó R [2] = 1, para 2; nó R [3] = 3, para 3; e nó R [4] = 4, para 4;

Vamos dar um exemplo maior para entender o resultado da reordenação:

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

O grau do nó 'A' é definido como o número de nós conectados a 'A'.

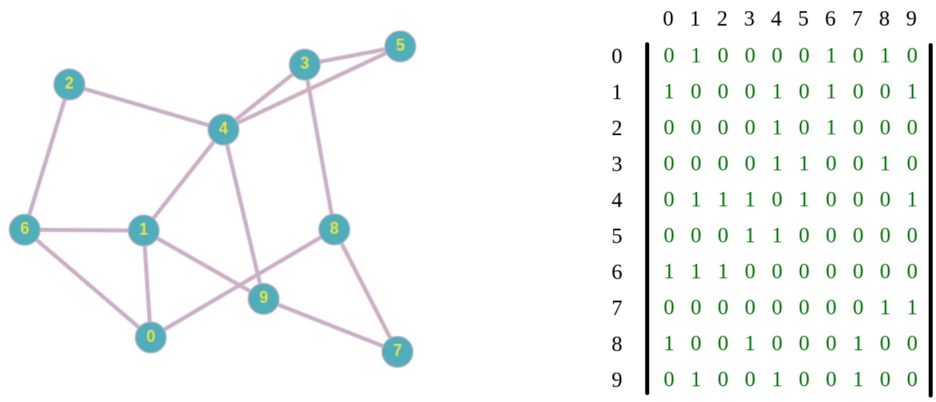
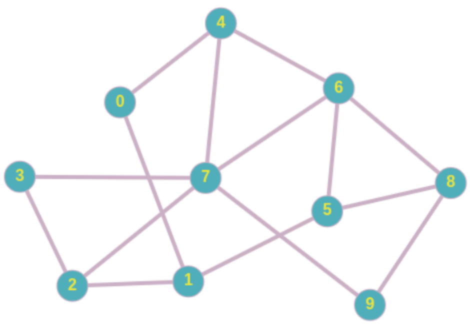


Figura 2. Matriz de adjacência

Ordem de permutação de objetos (R): $7 \quad 8 \quad 3 \quad 5 \quad 9 \quad 0 \quad 1 \quad 4 \quad 6 \quad 2$

Aplicada a conversão do nó R[i] para o nó i.

Assim, o gráfico se torna:



$$
\begin{array}{c|cccccccccc|}
 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
3 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
4 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
5 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
6 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
7 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
8 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0
\end{array}
$$

Figura 3. Matriz convertida

O resultado da reorganização pode ser visto pela matriz de adjacência dos dois gráficos:

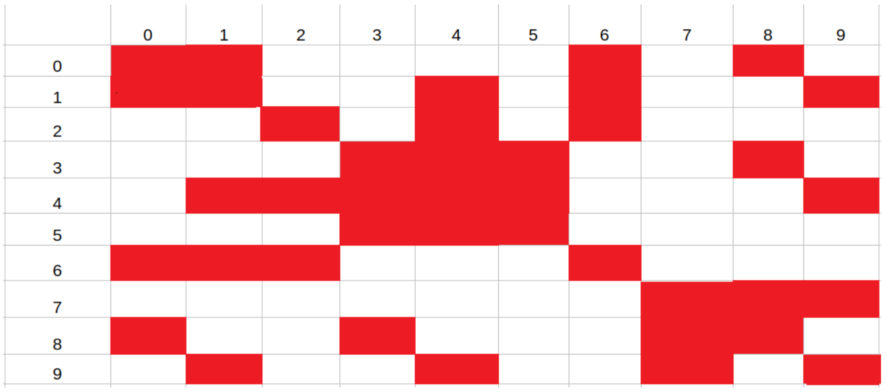


Figura 4. Matriz original

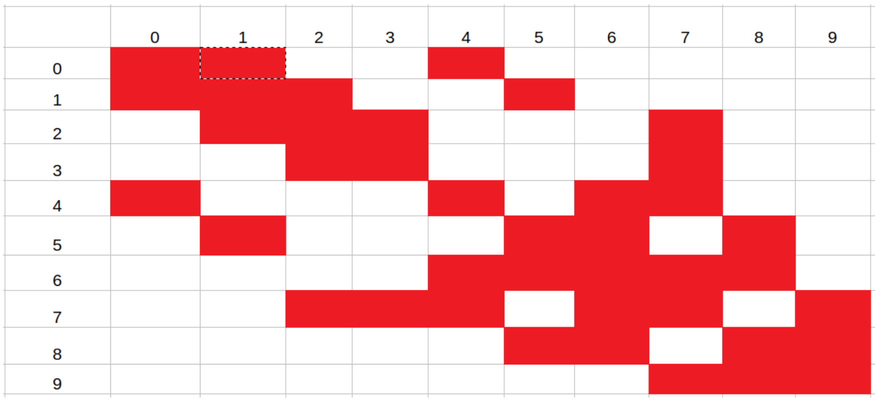


Figura 5. Matriz Reordenada RCM

Pode-se observar que os elementos estão agrupados em distâncias menores, logo o algoritmo de Cuthill-Mckee ajuda na reordenação de uma matriz quadrada em uma matriz não distribuída [RCM].

3. Experimentos RCM

Para realizar os experimentos com o algoritmo RCM foram escolhidos 4 grafos com diferentes tamanhos e com uma quantidade significativa de vértices e arestas. Para demonstrar o funcionamento de RCM foi utilizado um grafo pequeno para que se possa visualizar o processo de execução do algoritmo.

O Framework utilizado nos experimentos foi o NetworkX que já conta como uma implementação do RCM e uma heurística para escolher o vértice inicial que servirá de ponto de partida do RCM, a heurística escolhe o vértice de menor grau do grafo G.

O Grafo G = (V,E) com V = 39 e E = 39 é mostrado abaixo:

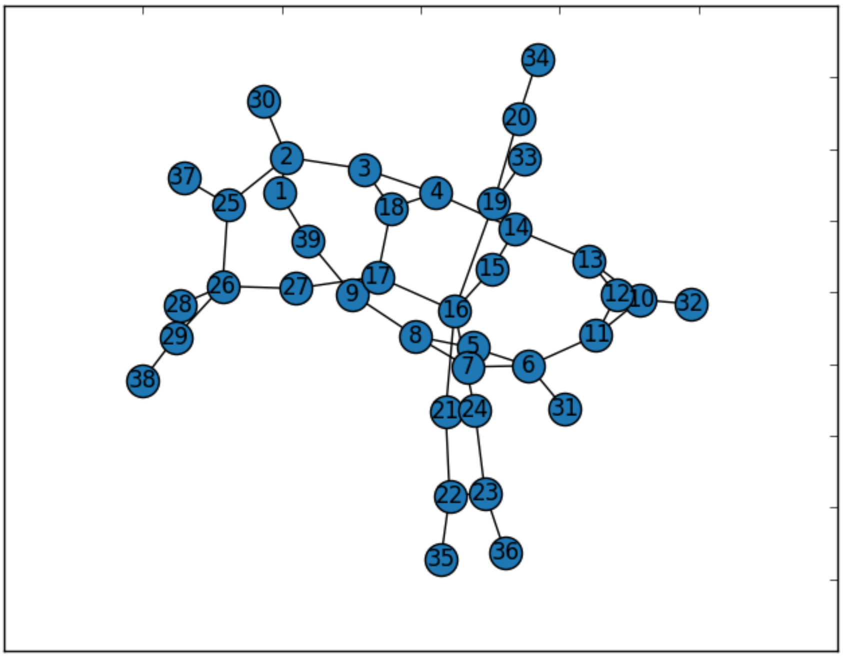


Figura 6. Grafo G

A saída do algoritmo RCM é uma lista com a nova ordem dos vértices.Para o grafo G o RCM retorna a seguinte saída que chamaremos de P:

P = [34, 35, 36, 31, 32, 11, 20, 33, 22, 23, 6, 10, 12, 19, 21, 24, 7, 5, 38, 13, 15, 16, 8, 29, 28, 27, 14, 17, 9, 26, 37, 4, 18, 39, 25, 3, 1, 2, 30]

Após obter a lista P o grafo G tem seus vértices permutados seguindo a ordem definida em P, chamaremos o grafo com os vértices permutados de H.

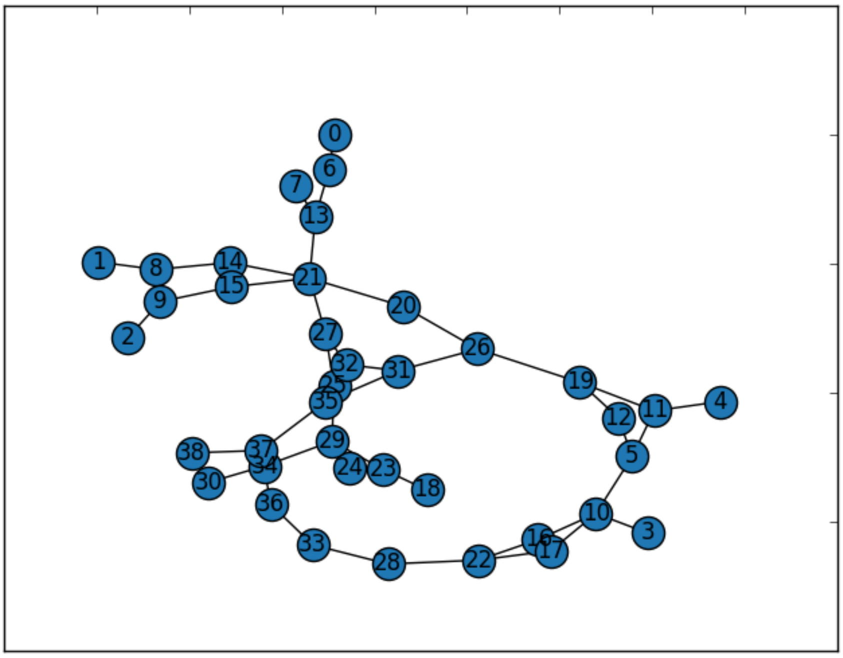


Figura 7. Grafo H permutado

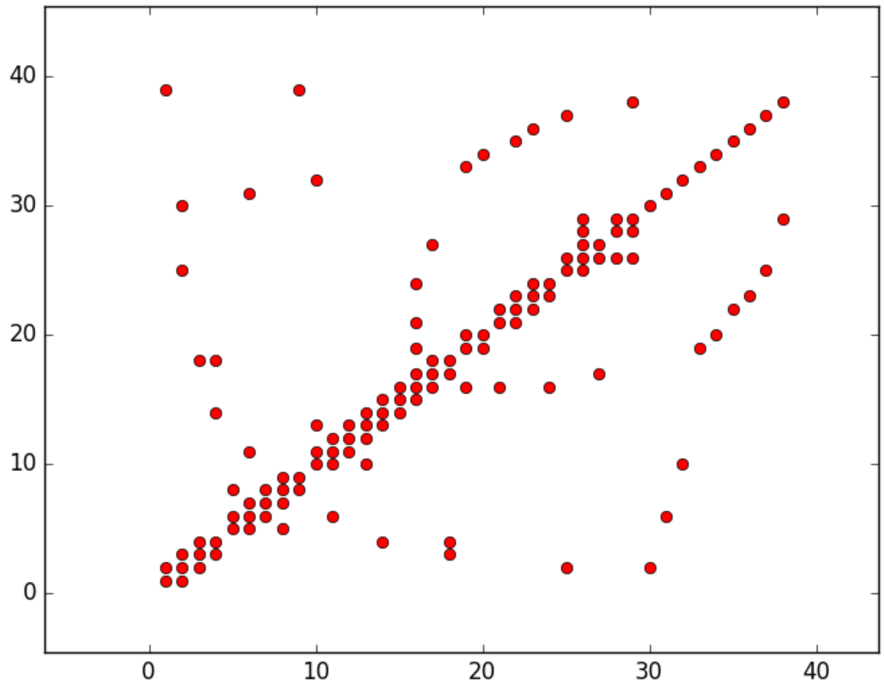


Figura 8. Representação do Grafo G no plano cartesiano

Abaixo mostraremos o grafo H representado no plano cartesiano com os vértices permutados. Note que após a troca dos vértices os elementos que antes estavam distantes da diagonal principal agora estão mais próximos da diagonal principal, reduzindo a largura de banda do grafo.

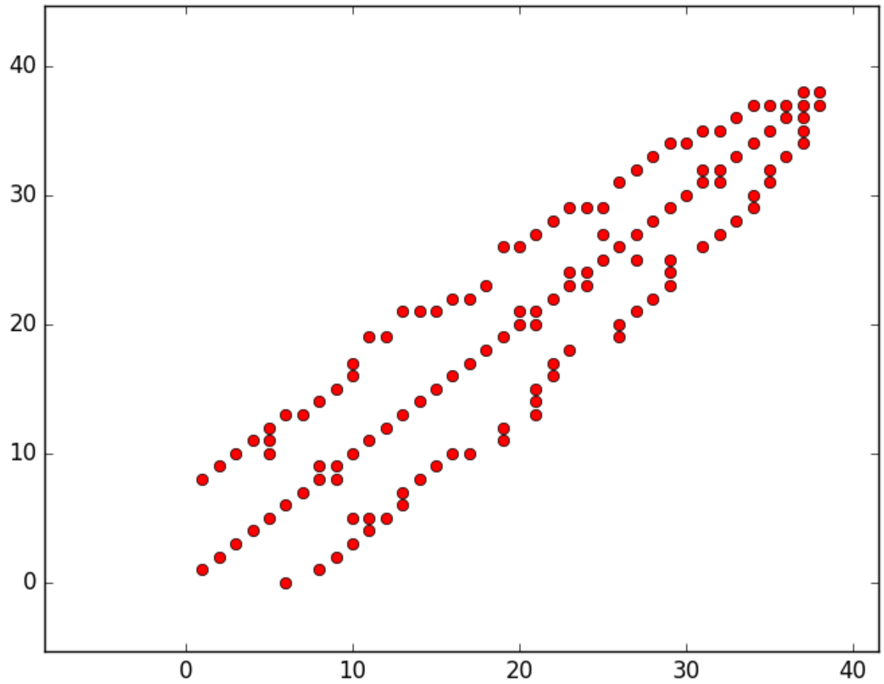


Figura 9. Representação do Grafo H no plano cartesiano

3.1. Execução do RCM nos demais grafos

Abaixo mostraremos a execução do RCM em 4 grafos de diferentes tamanhos, com **V** número de vértices e **E** número de arestas.

Grafo 1: V = 118, E = 118

Grafo 2: V = 245, E = 245

Grafo 3: V = 445, E = 445

Grafo 4: V = 1138, E = 1138

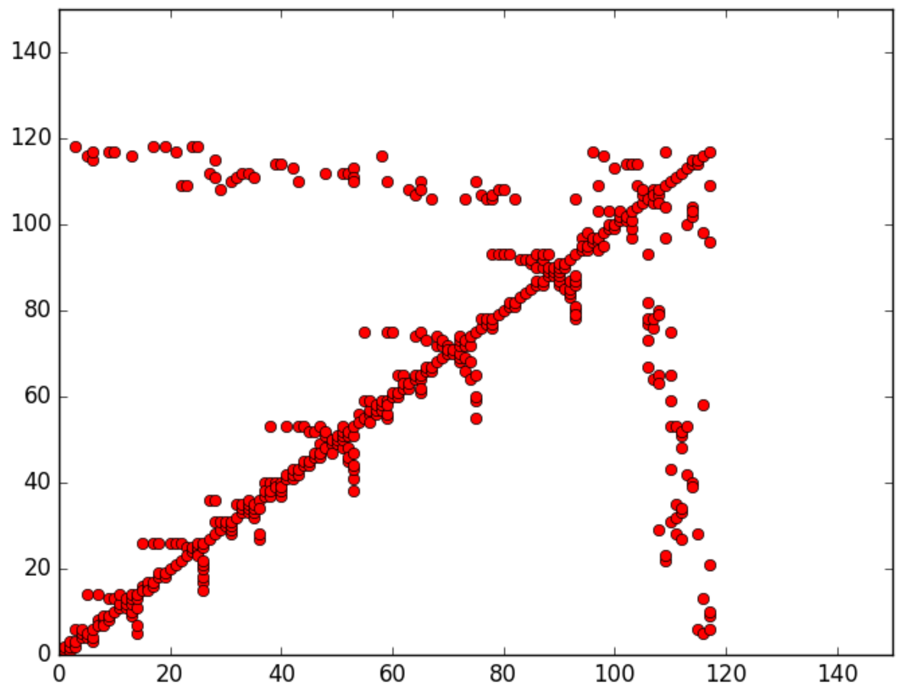


Figura 10. Representação do Grafo 1 no plano cartesiano

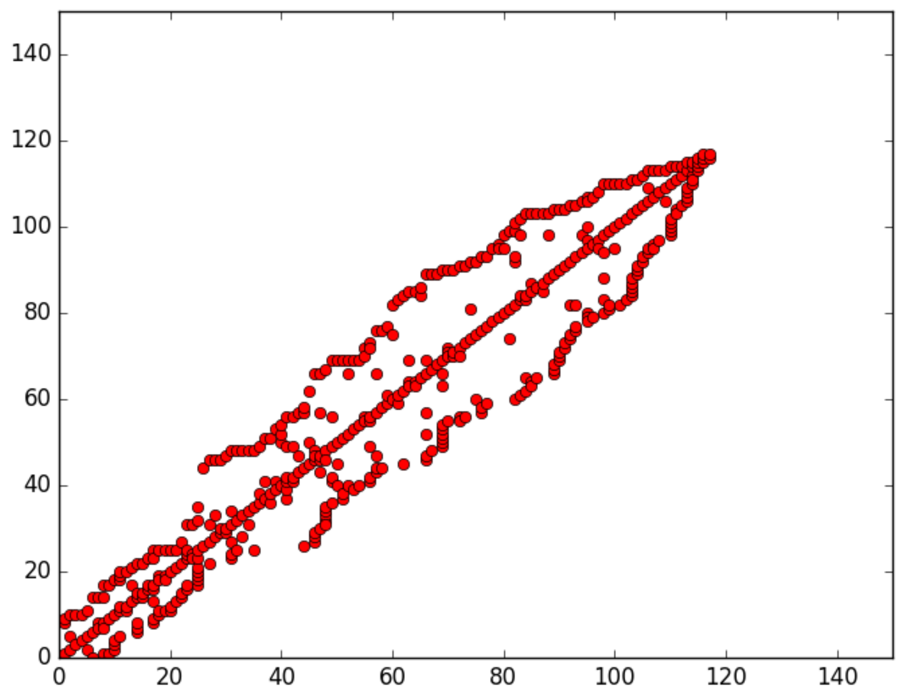


Figura 11. Representação do Grafo 1 no plano cartesiano após execução do RCM

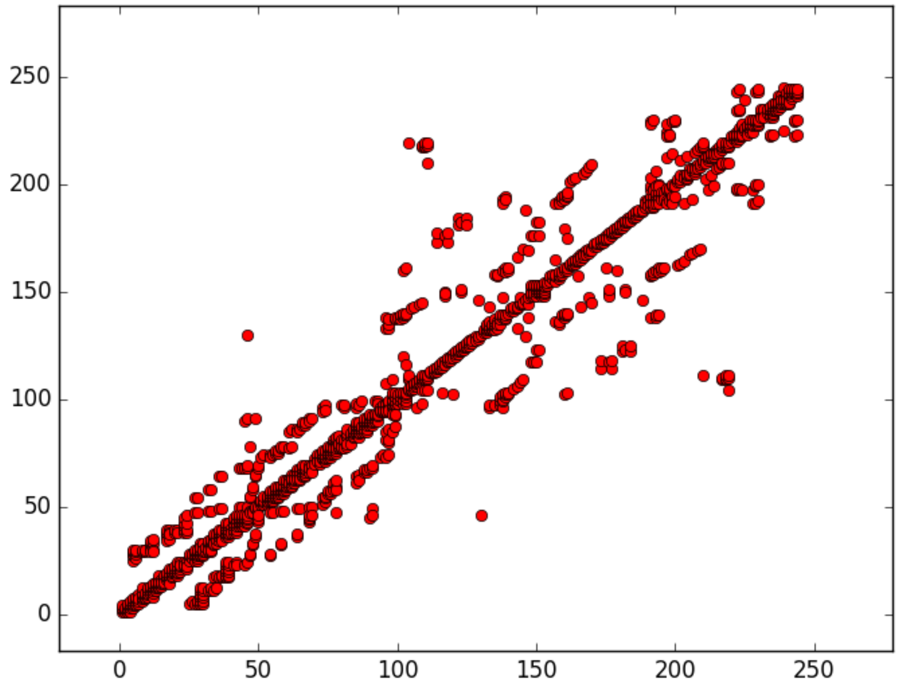


Figura 12. Representação do Grafo 2 no plano cartesiano

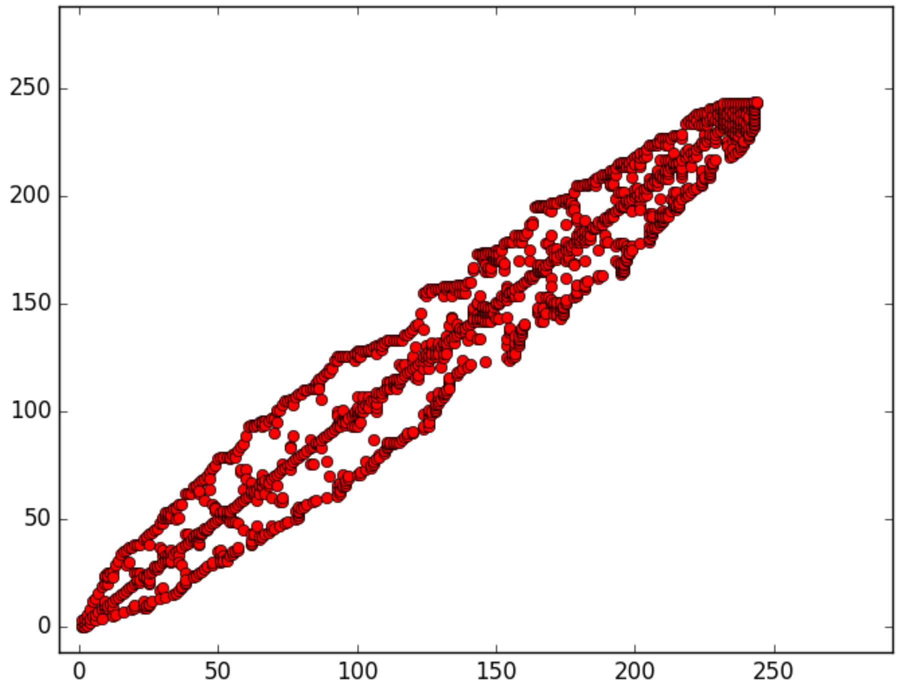


Figura 13. Representação do Grafo 2 no plano cartesiano após execução do RCM

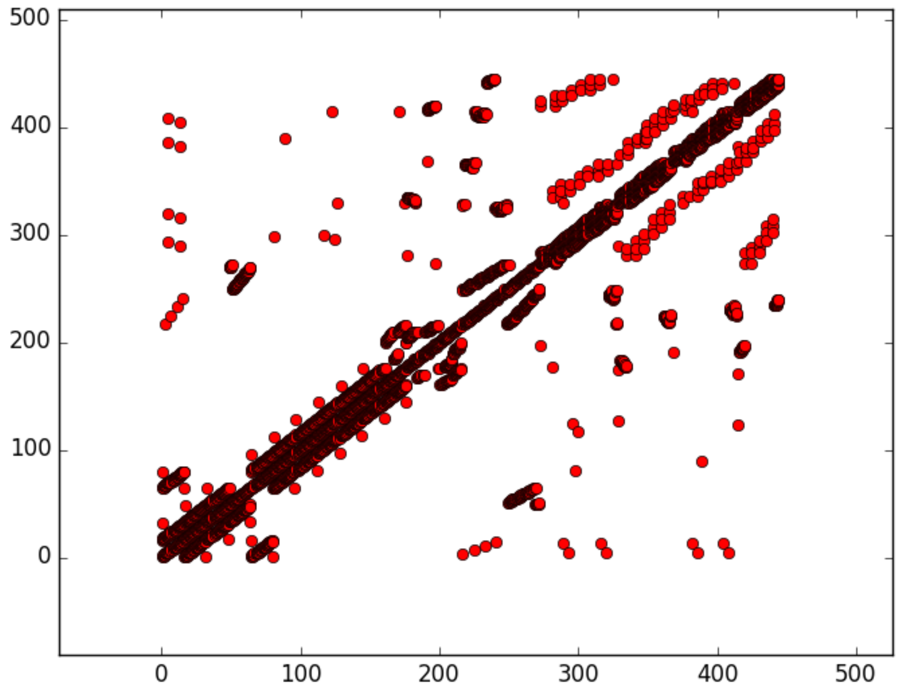


Figura 14. Representação do Grafo 3 no plano cartesiano

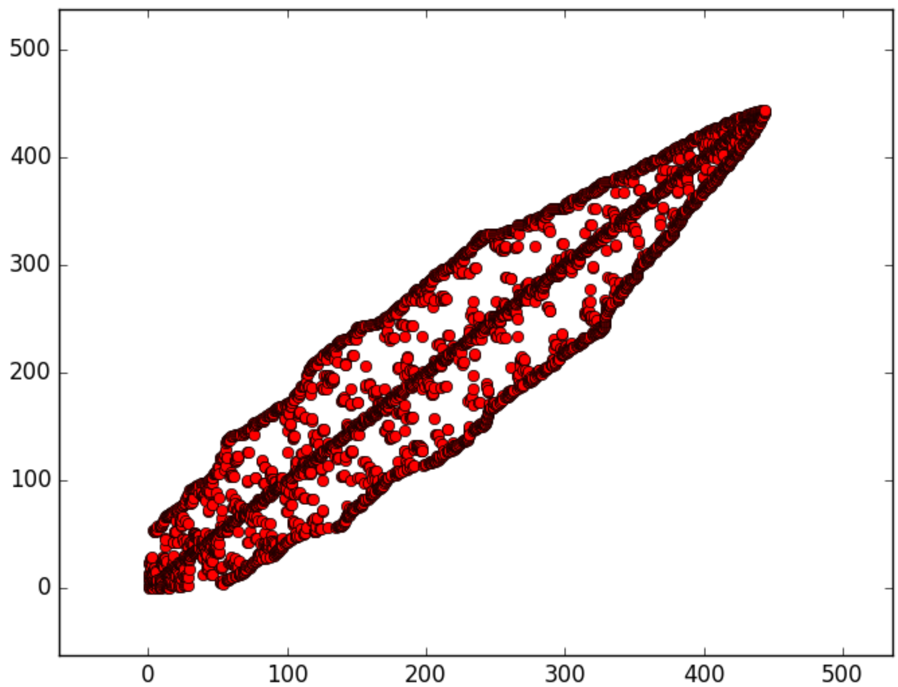


Figura 15. Representação do Grafo 3 no plano cartesiano após execução do RCM

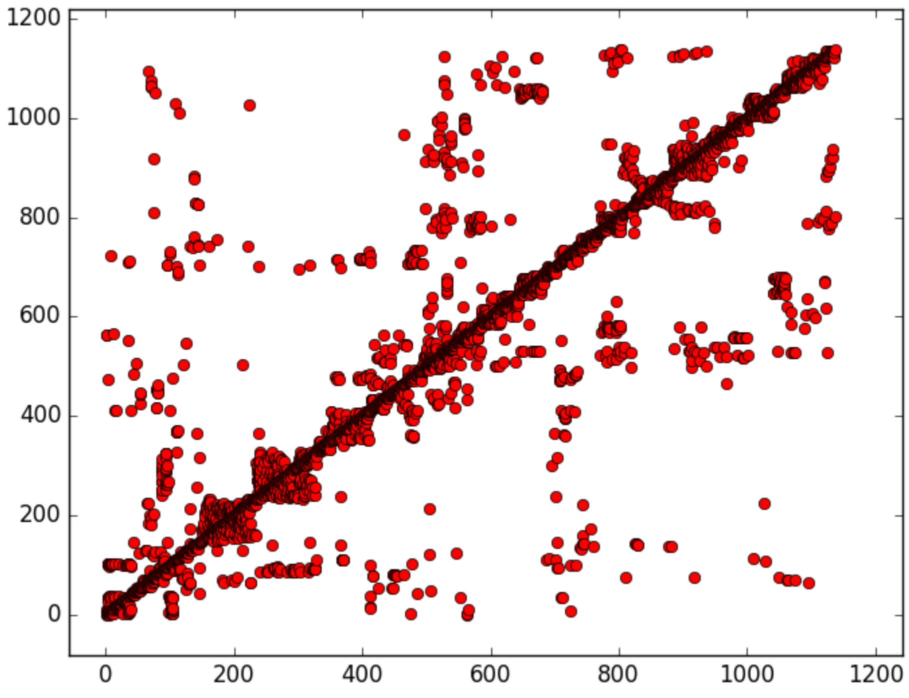


Figura 16. Representação do Grafo 4 no plano cartesiano

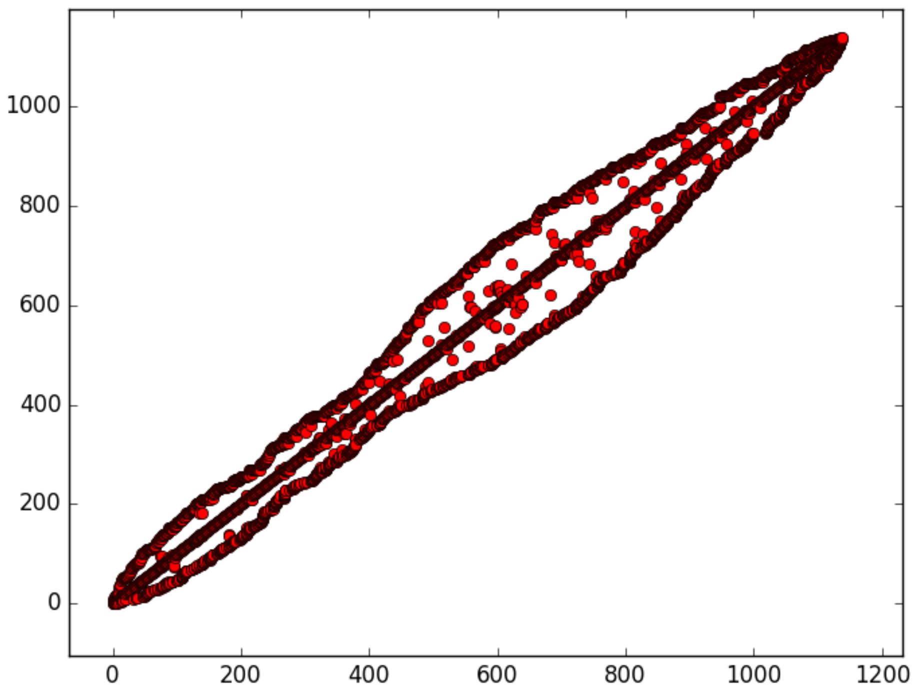


Figura 17. Representação do Grafo 4 no plano cartesiano após execução do RCM

4. Algoritmo Genético

Algoritmos genéticos referem-se a uma família de modelos computacionais inspirado pela evolução. Esses algoritmos codificam uma solução potencial para um problema específico em uma estrutura e aplicam operações de recombinação com intenção de preservar informações críticas[Whitley 1993]. São desenvolvidos baseados em comportamentos naturais. O algoritmo seleciona os indivíduos mais aptos de uma população a reproduzirem e passar seus genes para as próximas gerações, tal qual acontece na teoria da evolução de Darwin. Assim a cada geração a população se torna mais apta para determinada situação.

O algoritmo apresentado é uma adaptação de um algoritmo genético, uma meta-heurística que foi desenvolvida na plataforma de desenvolvimento em nuvem *Google Colaboratory*, na linguagem de programação *Python*. Utilizou-se as bibliotecas auxiliares *Networkx* para utilização do algoritmo Cuthill-Mckee reverso e operações com grafos, *Matplotlib* para gerar as imagens e bibliotecas matemáticas para otimização de operações.

4.1. Principais Componentes

Os componentes fundamentais de um algoritmo genético podem ser divididos em partes, abaixo apresenta-se cada uma, no contexto do algoritmo desenvolvido:

- **Função Objetivo:** é a função que calcula a largura de banda máxima de uma matriz que deve ser minimizada.

- **Indivíduo:** é uma matriz com a largura de banda reduzida, gerada aleatoriamente a partir da saída do algoritmo RCM (Cuthill-Mckee reverso) que recebeu a matriz inicial a ser minimizada ou um descendente de uma matriz que apresentou bom resultado para o problema.

- **População:** é um conjunto de indivíduos. Em cada geração os descendentes dos melhores indivíduos tendem a continuar na população.

- **Seleção:** é a etapa em que os indivíduos da população são selecionados aleatoriamente para reproduzir ou não. Ocorre de acordo com o atributo desejado, no caso, os que têm menor valor de largura de banda tem maiores chances de reproduzir. Nem todos os indivíduos reproduzem.

- **Mutação:** essa etapa foi desenvolvida dentro da Reprodução, e tem a finalidade de aumentar a variabilidade da população.

- **Reprodução:** é a parte onde a população é melhorada. Indivíduos que passaram na seleção e estão aptos a reproduzir são comparados em pares. Um número fracionário aleatório entre 0 e 1 é gerado, se este for menor ou igual à taxa de mutação o pior indivíduo do par, com maior largura de banda será mantido na população. Se o número for maior que a taxa de mutação, o indivíduo que apresentar a menor largura de banda do par é o melhor, com gene dominante. Então o RCM recebe como entrada a matriz de adjacência do indivíduo dominante, é executado e retorna uma nova matriz que representa um descendente do melhor indivíduo.

4.2. Lógica do algoritmo

1. Faz-se a definição dos parâmetros de controle, **Tamanho da população**, **Número de casais**, **Número de iterações** e **Taxa de mutação**.

2. A partir do *upload* da matriz a ser minimizada gera-se a matriz de adjacência inicial, calculando os pesos de cada aresta a partir da diferença de valor dos nós adjacentes.

3. Cria-se a partir da matriz inicial, a população inicial, executando *n* vezes, com início em um nó aleatório, o algoritmo de Cuthill-Mckee reverso até atingir o **Tamanho da população**.

4. Faz a iteração *m* vezes, de acordo com o **Número de iterações** especificado.

    (a) Cria-se uma nova população vazia.

    (b) Calcula-se a largura de banda de todos os indivíduos da população inicial.

    (c) Seleciona-se os indivíduos que vão reproduzir.

    (d) Popula-se a nova população cruzando os indivíduos de acordo com o **Número de casais**.

    (e) Adiciona-se à nova população indivíduos gerados a partir da matriz de adjacência inicial até atingir o **Tamanho da população** inicial.

    (f) Substitui-se os indivíduos da população inicial pelos da nova população.

5. Ao final das iterações tem-se a melhor matriz, que possui a menor largura de banda encontrada.

5. Grafos após a execução do algoritmo genético

O grafo é passado como entrada para o algoritmo genético para a redução da largura de banda. Abaixo situam-se os grafos anteriores após a execução do algoritmo genético, com suas respectivas larguras de banda e o tempo total de execução:

- A largura de banda inicial do grafo G é de 38

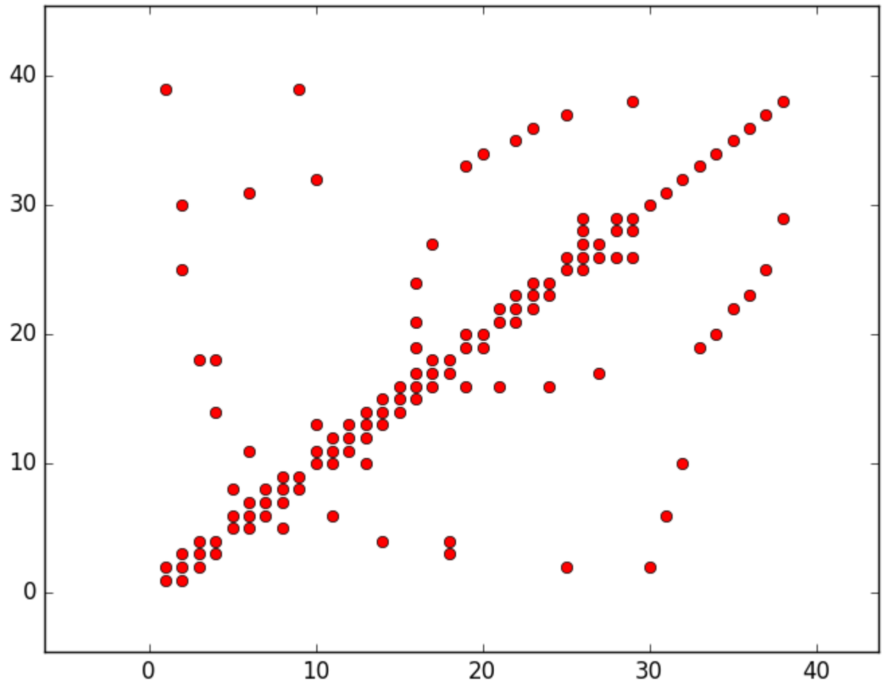


Figura 18. Representação do Grafo G no plano cartesiano

```
Melhor valor de largura de banda encontrado:  5.0

Tempo de execucao:  49.33166575431824s
```

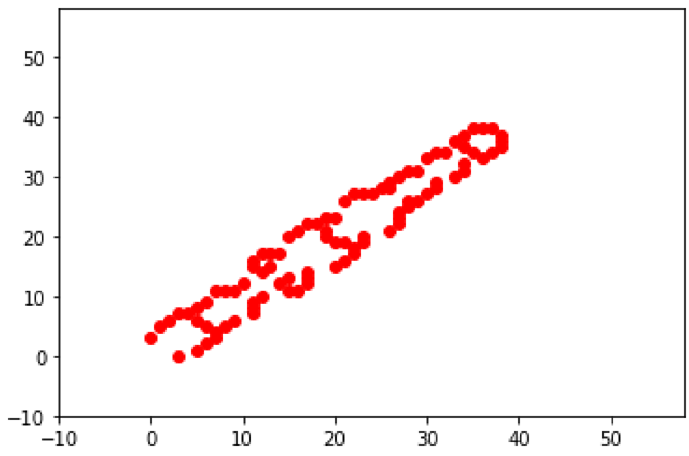


Figura 19. Representação do Grafo G no plano cartesiano após a execução do algoritmo genético

- A largura de banda inicial do grafo 1 é de 115

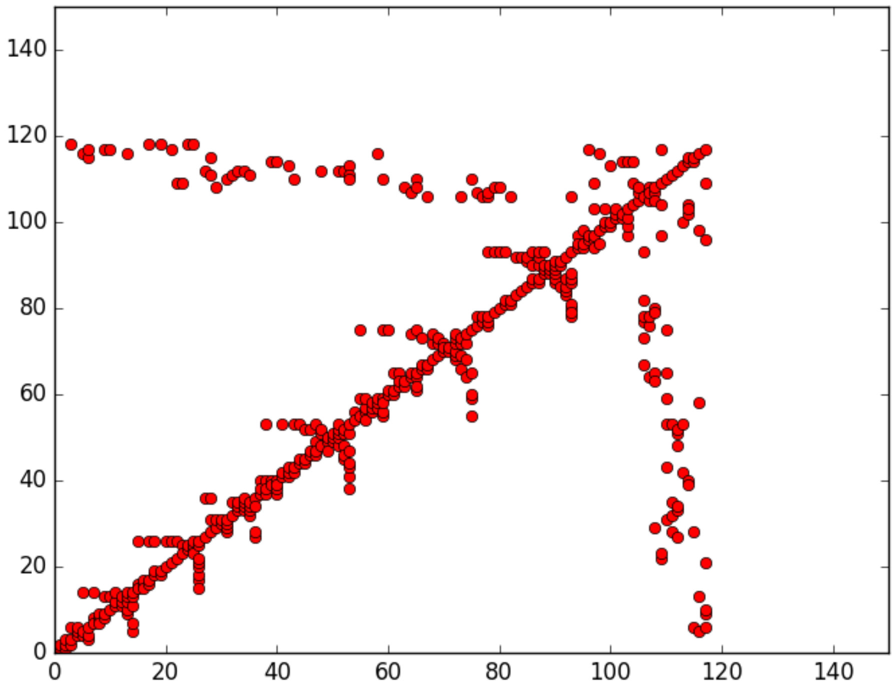


Figura 20. Representação do Grafo 1 no plano cartesiano

```
Melhor valor de largura de banda encontrado:  16.0

Tempo de execucao:  383.66080141067505s
```

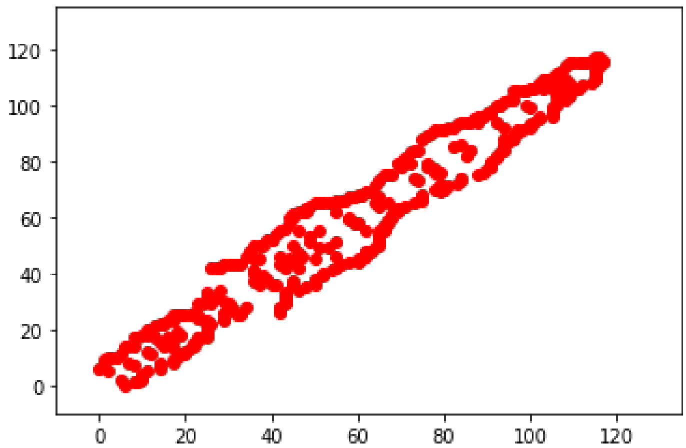


Figura 21. Representação do Grafo 1 no plano cartesiano após a execução do algoritmo genético

- largura de banda inicial do grafo 2 é de 115

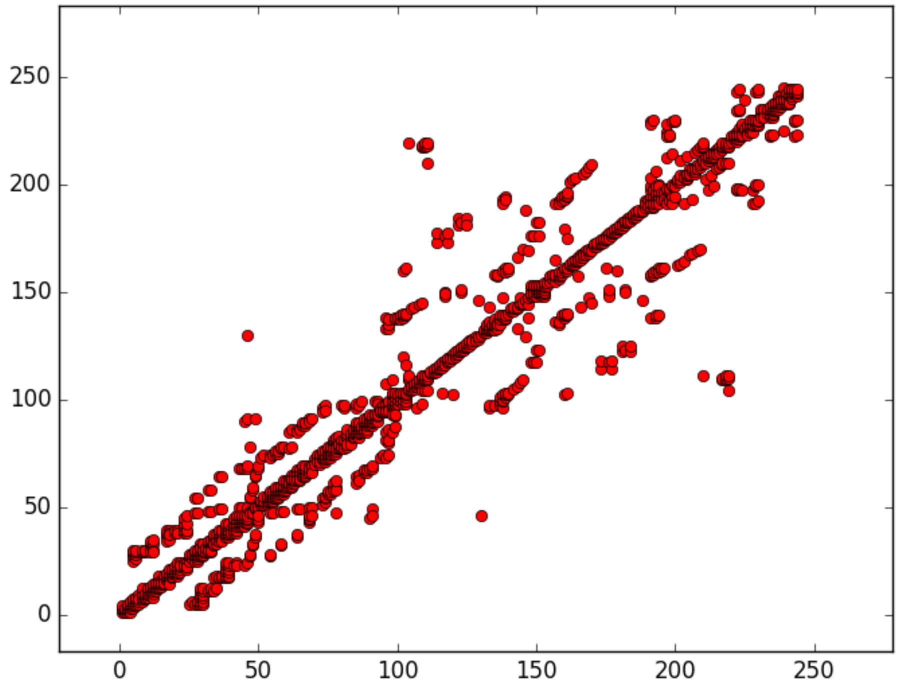


Figura 22. Representação do Grafo 2 no plano cartesiano

```
Melhor valor de largura de banda encontrado:  32.0

Tempo de execucao:  1641.4333894252777s
```

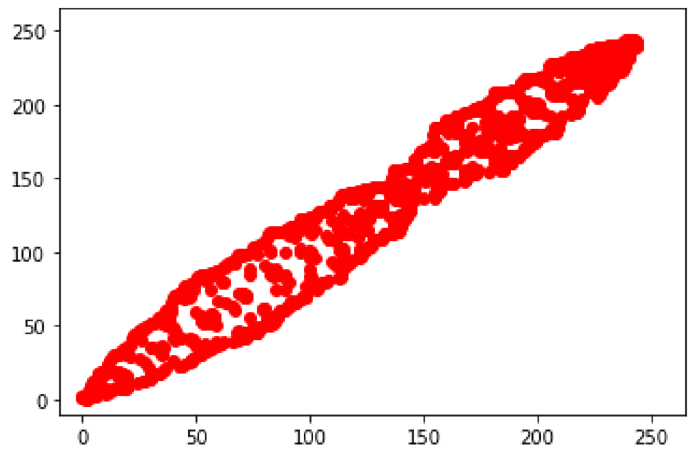


Figura 23. Representação do Grafo 2 no plano cartesiano após a execução do algoritmo genético

- A largura de banda inicial do grafo 3 é de 405

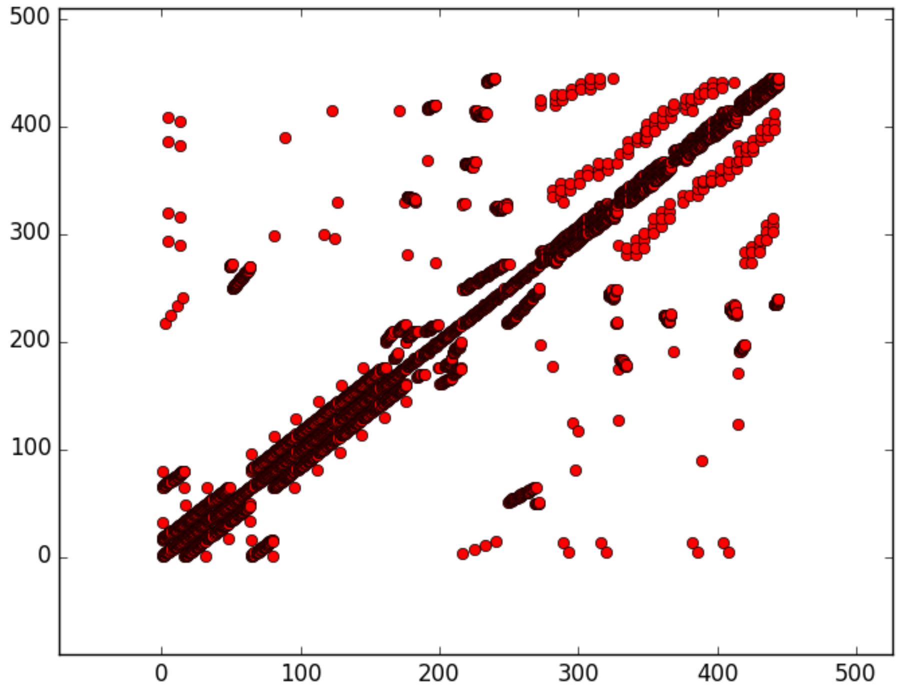


Figura 24. Representação do Grafo 3 no plano cartesiano

```
Melhor valor de largura de banda encontrado:  72.0

Tempo de execucao:  5364.230580091476s
```

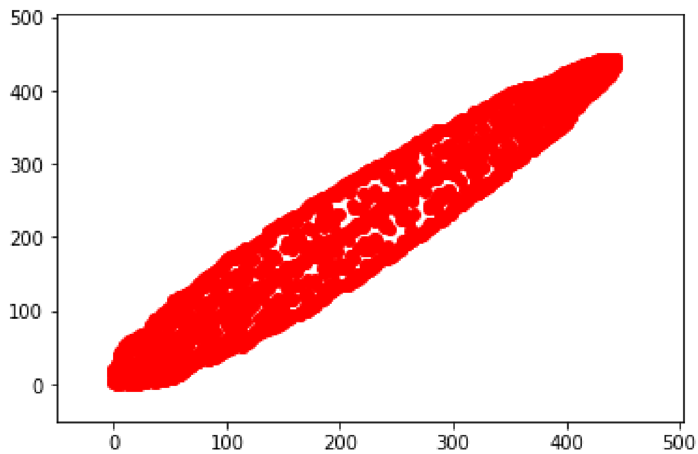


Figura 25. Representação do Grafo 3 no plano cartesiano após a execução do algoritmo genético

- A largura de banda inicial do grafo 4 é de 1030

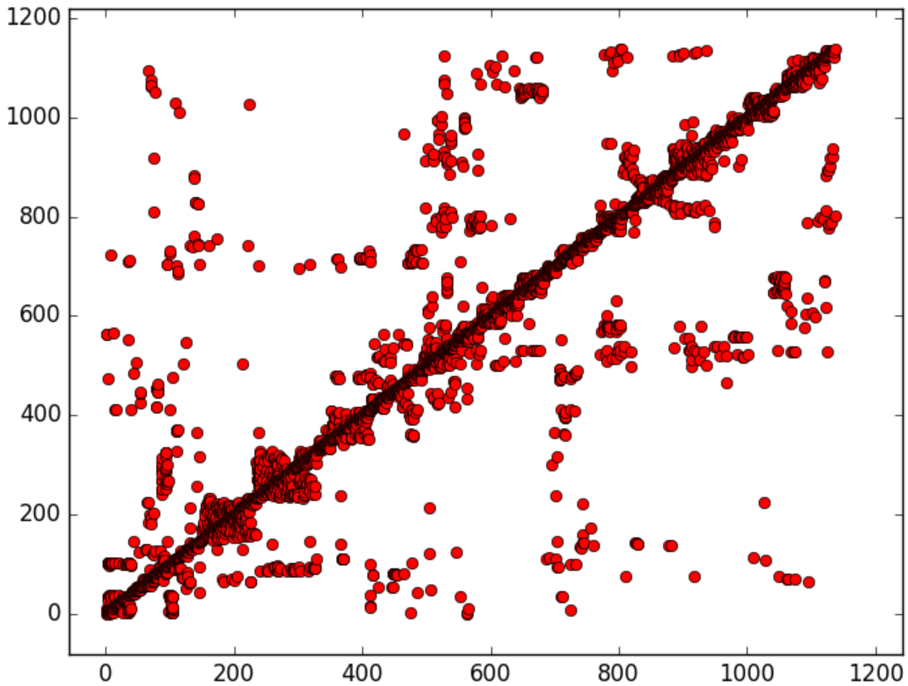


Figura 26. Representação do Grafo 4 no plano cartesiano

```
Melhor valor de largura de banda encontrado:  110.0

Tempo de execucao:  27702.767169475555s
```

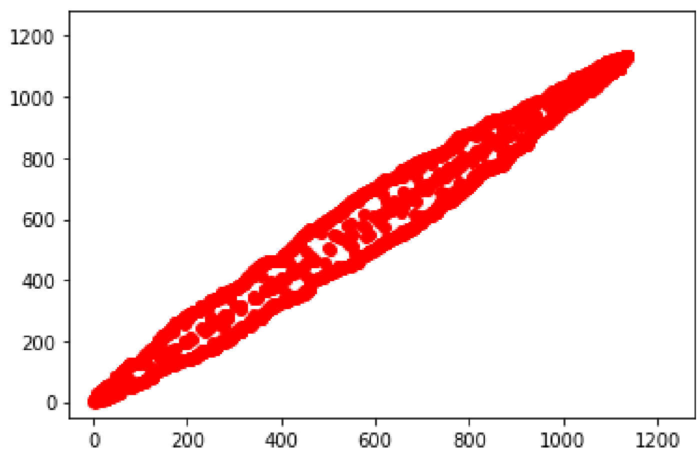


Figura 27. Representação do Grafo 4 no plano cartesiano após a execução do algoritmo genético

6. Conclusão

O desenvolvimento do trabalho proporcionou vários desafios para sua realização entre eles o mais significativo foi a escolha da heurística que seria utilizada pra a redução da largura de banda do grafo, visto que a escolha da heurística correta é essencial para a obtenção de bons resultados.

O experimento foi feito em duas etapas, a primeira consistiu no pré-processamento do grafo com o algoritmo RCM, que fornece uma boa solução inicial para o problema visto que o algoritmo sozinho foi capaz de reduzir significativamente a largura de banda do grafo.

A segunda etapa do desenvolvimento consistiu na utilização do algoritmo genético que com a matriz inicial como entrada cria uma população aleatória de matrizes a partir da execução do RCM (Reverse Cuthill-Mckee) e este aprimora a redução da largura de banda pela seleção dos melhores indivíduos (matrizes) concentrando o maior número possível de vértices próximos a diagonal principal do grafo.

No final do experimento foi possível observar que todos os grafos testados tiveram suas respectivas larguras de banda reduzidas em mais de 70%. O principal ponto negativo do algoritmo é o tempo de execução, visto que a criação da população inicial e das gerações futuras possui um custo alto de processamento. Em matrizes grandes o tempo de processamento pode se tornar inviável, porém, é possível ajustar os parâmetros de execução, assim otimizando o processamento e as respostas.

A realização deste trabalho proporcionou um enorme aprendizado sobre heurísticas e dos vários *Frameworks* que as disponibilizam. Demandou bastante tempo, dedicação e estudo da dupla para entender seus funcionamentos e a forma correta de utilizá-los para obter os resultados desejados no projeto.

Referências

Geeksforgeeks - rcm (reverse cuthill mckee algorithm).

E. Cuthill, J. M. (1969). *Reducing the bandwidth of sparse symmetric matrices*. Proceedings of the 1969 24th.

W. Liu, A. H. S. (2006). Comparative analysis of the cuthill-mckee and the reverse cuthill-mckee ordering algorithms for sparse matrices. In *SIAM J. Numer. Anal.*, pages 198–213. Society for Industrial and Applied Mathematics.

Whitley, D. (1993). *A Genetic Algorithm Tutorial*. Technical Report CS-93-103.