UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
LICENCIATURA EM MATEMÁTICA

GUILHERME BARBOSA DOS SANTOS

MÉTODOS ESPECTRAIS DE PARTIÇÃO DE GRAFOS

CURITIBA
2018

GUILHERME BARBOSA DOS SANTOS

MÉTODOS ESPECTRAIS DE PARTIÇÃO DE GRAFOS

Trabalho de Conclusão de Curso apresentado à Banca Examinadora como requisito parcial para obtenção do título de licenciado em Matemática pela Universidade Tecnológica Federal do Paraná sob a orientação do Prof. Doutor João Luis Gonçalves.

CURITIBA
2018

Ministério da Educação
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
Câmpus Curitiba
Diretoria de Graduação e Educação Profissional
Departamento Acadêmico de Matemática
Coordenação do Curso de Licenciatura em Matemática

TERMO DE APROVAÇÃO

"MÉTODOS ESPECTRAIS DE PARTIÇÃO DE GRAFOS"

por

"Guilherme Barbosa dos Santos"

Este Trabalho de Conclusão de Curso foi apresentado às **11h10** do dia **4** de **dezembro** de 2018 na sala **a103** como requisito parcial à obtenção do grau de Licenciado em Matemática na Universidade Tecnológica Federal do Paraná - UTFPR - Câmpus Curitiba. O(a) aluno(a) foi arguido pela Banca de Avaliação abaixo assinados. Após deliberação, de acordo com o parágrafo 1º do art. 37 do Regulamento Específico do trabalho de Conclusão de Curso para o Curso de Licenciatura em Matemática da UTFPR do Câmpus Curitiba, a Banca de Avaliação considerou o trabalho **aprovado** (aprovado ou reprovado).

_____ Prof.(a) Dr.(a) João Luis Gonçalves **(Presidente - UTFPR/Curitiba)**	_____ Prof.(a) Dr.(a) Francisco I. Secolo Ganacim **(Avaliador 1 - UTFPR/Curitiba)**
_____ Prof.(a) Dr.(a) Roy Wilhelm Probst **(Avaliador 2 - UTFPR/Curitiba)**	
_____ Profª Drª Priscila Savulski Ferreira de Miranda **(Professor Responsável pelo TCC – UTFPR/Curitiba)**	_____ Profª Drª Neusa Nogas Tocha **(Coordenador do curso de Licenciatura em Matemática – UTFPR/Curitiba)**

Este trabalho é dedicado à minha família. Sem eles eu não teria alcançado isso.

Resumo

SANTOS, Guilherme Barbosa dos. **Métodos Espectrais de Partição de Grafos**. 2018. 44 f. Trabalho de Conclusão de Curso (Graduação) – Licenciatura em Matemática. Universidade Tecnológica Federal do Paraná. Curitiba, 2018.

Palavras-chave: grafos, comunidades, partição.

Resumo:
A Teoria de Grafos tem grande potencial para a descrição e modelagem matemática de fenômenos associados a redes. Em problemas dessa natureza uma dificuldade recorrente é o tamanho da rede, ou seja, o número de vértices e arestas. Para contornar essas dificuldades, uma alternativa é decompor o grafo associado ao problema, ou ainda agrupar os vértices com propriedades semelhantes, gerando assim um novo grafo com menos vértices mas que mantém uma representação adequada do fenômeno. Esses agrupamentos de vértices serão denominados de comunidades. Outro aspecto, que não pode ser negligenciado, é a boa apresentação dos dados que os grafos oferecem. Nesse sentido a detecção de comunidades tem papel de sintetizar ainda mais essas informações. Contudo a detecção de comunidades raramente pode ser feita empiricamente, principalmente para grafos grandes. Portanto, faz-se necessário um tratamento analítico do grafo, com rigor matemático. Um tratamento matematicamente rigoroso é um ponto positivo pois exigirá o uso de teorias mais desenvolvidas, como a álgebra linear por exemplo. Assim, existirá uma quantidade maior de ferramentas para abordar um tema que não nos é tão familiar. Este trabalho tem como objetivo apresentar a maximização da modularidade e a minimização da função Corte, usando análise espectral do grafo, como uma alternativa à partição ou decomposição de grafos.

Abstract

SANTOS, Guilherme Barbosa dos. **Spectral Methods of Graph Partitioning**. 2018. 44 f. Monography - Bachelor in Mathematics. Federal Technological University of Paraná.Curitiba, 2018.

Keywords: graphs, communities, partition.

Summary:
The Theory of Graphs has great potential for the description and mathematical modeling of phenomena associated to networks. In problems of this nature a recurring difficulty is the size of the network, i.e. the number of vertices and edges. To overcome these difficulties, an alternative is to decompose the graph associated with the problem by grouping the vertices with similar properties, thus generating a new graph with fewer vertices but still representing the same phenomenon. These groupings of vertices will be called communities. Another aspect, that cannot be overlooked, is the good presentation of data that graphs offer. In this context, the detection of communities has the role of synthesizing this information even further. However, community detection can rarely be done empirically, especially for large graphs. Therefore, an analytical treatment of the graph is made necessary , with mathematical rigor. This mathematically rigorous treatment is a positive point because it will require the use of more developed theories, such as linear algebra. Thus, we will have a greater amount of tools to approach a theme that might not be familiar to us. This work aims to present the maximization of modularity and the minimization of the cutting function, using spectral analysis of the graph as an alternative to partitioning or decomposition of graphs.

Conteúdo

Introdução

Dentre as várias áreas de pesquisa em Matemática, destaca-se a Matemática Discreta, e em particular, a Teoria de Grafos, que estuda como a informação é compartilhada entre agentes de uma rede. Essas redes podem ser de comunicação, de transportes e redes sociais dentre outras. Assim pode-se mensurar o grande potencial de aplicação da Teoria de Grafos para a modelagem de problemas em redes.

A Teoria de Grafos, principalmente por sua análise algébrica e espectral, proporciona ferramentas fundamentais para o estudo de modelos em redes. Para isso, é necessário ter como base alguns outros conteúdos matemáticos, em particular aqueles contemplados em um curso de Licenciatura em Matemática, como por exemplo, Álgebra Linear, Análise Combinatória, Métodos Numéricos, Equações Diferenciais e Noções de Computação.

Diante do potencial e da crescente demanda por soluções de problemas envolvendo redes, justifica-se este trabalho. Além do estudo sobre o tema, busca-se elaborar um texto introdutório sobre a classe de métodos espectrais em problemas modelados usando grafos, principalmente problemas sobre a detecção de comunidades em redes.

Para isso, no Capítulo 1 estão apresentadas algumas definições retiradas de [4] e [3] que dão a estruturação teórica ao trabalho, contendo resultados importantes para o desenvolvimento da classe de métodos a ser apresentada. Não serão citadas todas as definições relacionadas à teoria grafos por sua densidade. Serão abordados os conceitos básicos, com foco na parte essencial a análise espectral do Grafo.

O Capítulo 2 tem como objetivo explorar métodos de partição de grafos contidos em [5]. Os grafos, geralmente, podem representar redes, sejam elas neurais, de transporte ou sociais, entre outros exemplos. Um problema comum é a detecção de comunidades nessas redes. Uma ferramenta útil para a resolução desse tipo de problema é o particionamento de um grafo utilizando métodos algébricos e explorando o espectro de algumas das matrizes associadas aos grafos que serão apresentadas no Capítulo 1.

Experimentos numéricos que ilustram a aplicação dos métodos apresentados são discutidos no Capítulo 3.

No Capítulo 4 será feita a conclusão sobre o estudo da Teoria de Grafos e dos métodos utilizados para a detecção de comunidades e os resultados esperados futuramente, relacionando os métodos teóricos e o algoritmo de detecção de comunidades executado pelo software R.

Os códigos para a implementação dos métodos foram implementados em R e estão disponíveis no Apêndice deste trabalho.

Capítulo 1

Grafos

A Teoria de Grafos possui muitas definições e resultados que servem a resolução de diversos problemas em redes, por exemplo os mais conhecidos são os problemas de coloração e de caminho mínimo ou de custo mínimo. Neste capítulo serão expostos algumas definições acerca da Teoria de Grafos que constam em [4] e [3], principalmente as necessárias para o desenvolvimento do restante do trabalho.

1.1 Definições Importantes

Inicialmente, será apresentada a definição de grafo e as notações utilizadas ao longo do trabalho sobre essa teoria.

Definição 1.1.1. Um **grafo** é uma estrutura $G = G(V, E)$, formada por um conjunto finito e não vazio V cujos elementos são denominados **vértices**, e um outro conjunto E cujos elementos são pares ordenados de elementos de V, cada elemento de E é denominado **aresta**.

Definição 1.1.2. Define-se $|V|$, a **ordem** de G, como o número de vértices de um grafo $G = G(V, E)$ e, analogamente, define-se $|E|$, o **tamanho** de G, como o número de arestas de $G = G(V, E)$.

Definição 1.1.3. Se $u, v \in V$ e $e = \{u, v\} \in E$, então, diz-se que e **incide** em u e v.

Definição 1.1.4. Define-se **grau de um vértice** v, e denota-se por $d(v)$, como o número de arestas que incidem em v. Vértices ligados por arestas são ditos **vértices adjacentes**.

Definição 1.1.5. Quando V é um conjunto unitário, ou seja $|V| = 1$ e $E = \varnothing$, diz-se que G é o **grafo trivial**.

Exemplo 1.1.1. Para ilustrar as definições anteriores considere o grafo da Figura 1.1.

Pode-se observar que neste grafo o conjuntos dos vértices é $V = \{v_1, v_2, v_3, v_4, v_5\}$ e o conjunto das arestas é $E = \{(v_1, v_2), (v_2, v_3), (v_4, v_5), (v_1, v_4), (v_1, v_5), (v_2, v_5)\}$, resultando em ordem $|V| = 5$ e tamanho $|E| = 6$.

Os graus dos vértices são $d(v_1) = d(v_2) = d(v_5) = 3$, $d(v_3) = 1$ e $d(v_4) = 2$.

.

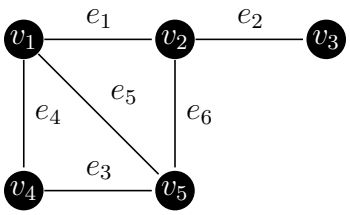

Figura 1.1: Grafo com 5 vértices 6 arestas.

Definição 1.1.6. Seja $G = G(V, E)$ um grafo. Se $G' = G'(V', E')$ é um grafo de forma que $V' \subset V$ e $E' \subset E$ denota-se $G' \subset G$ e diz-se que G' é um **subgrafo** de G. Quando $G' \subset G$ é tal que dois vértices são adjacentes em G' se e somente se eles são adjacentes em G, diz-se que G' é um **subgrafo induzido** de G. Um exemplo dssso é a Figura 1.2:

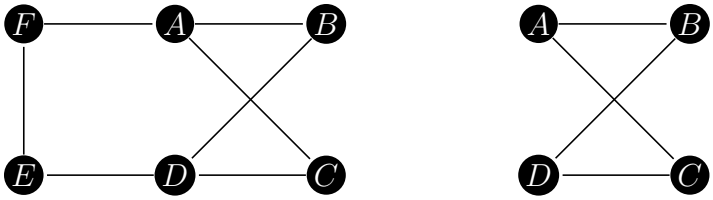

Figura 1.2: Um grafo e um subgrafo induzido

1.2 Alguns Tipos de Grafos

Agora, serão definidos alguns conceitos fundamentais sobre grafos, exemplificando cada definição de maneira simples.

Definição 1.2.1. Um **grafo regular de grau k ou k-regular** é um grafo em que todos os vértices têm o mesmo grau k. Pode-se observar essa definição na Figura 1.3.

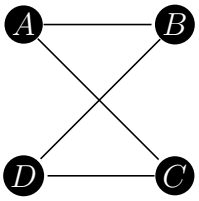

Figura 1.3: Um Grafo 2-regular.

Definição 1.2.2. Um **grafo completo** é um grafo no qual quaisquer dois vértices distintos são adjacentes.

O grafo completo com $n \geqslant 1$ vértices é $(n-1)$-regular e denotado por K_n. Observe um exemplo na Figura 1.4.

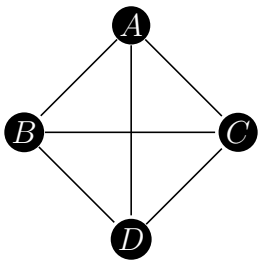

Figura 1.4: K_4.

Definição 1.2.3. Uma sequência finita $v_1, v_2, v_3 \cdots, v_k$ de vértices de um grafo $G(V, E)$ é dita uma **cadeia** de v_1 a v_k quando $\{v_i, v_{i+1}\} \in E$ para $1 \leq i \leq k-1$.

Definição 1.2.4. Diz-se que $v_1, v_2, \cdots, v_k$ é uma **cadeia fechada** quando $v_1 = v_k$ e uma **cadeia aberta** quando $v_1 \neq v_k$.

Definição 1.2.5. Um **caminho** é uma cadeia em que todos os vértices são distintos. Um caminho fechado é denominado ciclo.

Definição 1.2.6. O **comprimento** de um caminho ou de um ciclo é o número de arestas que ocorrem em cada um. O caminho e o ciclo com n vértices são denotados, respectivamente, por P_n e C_n. Em particular, o ciclo C_3 é chamado triângulo, ver Figura 1.5.

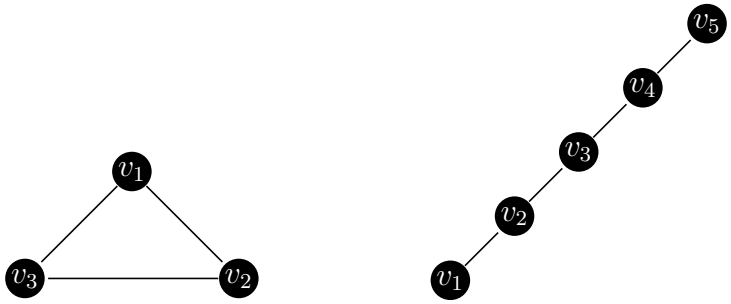

Figura 1.5: C_3 e P_5.

Definição 1.2.7. Grafo Conexo é um grafo no qual existe um caminho ligando cada par de vértices. Em caso contrário, o grafo é denominado **desconexo**.

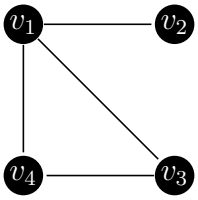

Figura 1.6: Grafo Conexo

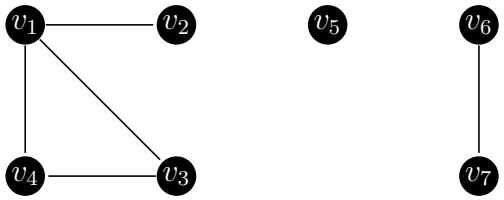

Figura 1.7: Grafo Desconexo

Definição 1.2.8. Um **grafo k-partido** é um grafo $G = G(V, E)$ no qual existe uma partição do conjunto de vértices V em k subconjuntos não vazios e disjuntos dois a dois, isto é,

$$V = Y_1 \cup Y_2 \cup Y_3 \cup \cdots \cup Y_k,$$

com $Y_i \cap Y_j = \emptyset, \quad \forall\, i \neq j$, de modo que as arestas de G sejam da forma $\{p, q\}, p \in Y_i, q \in Y_j$. Em outras palavras, não há vértices adjacentes em um mesmo subconjunto da partição.

Notação : Quando $k = 2$, chama-se o grafo de **bipartido**. Quando $k = 3$, chama-se o grafo de **tripartido**, e assim por diante. Um exemplo de um grafo bipartido é a Figura 1.8:

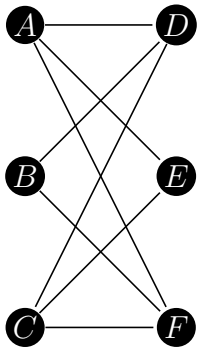

Figura 1.8: Grafo Bipartido.

Definição 1.2.9. Um **grafo bipartido completo** é um grafo $G = G(V_1 \cup V_2, E)$ bipartido em que cada vértice de V_1 é adjacente a todo vértice de V_2. Se $|V_1| = r$ e $|V_2| = s$, escreve-se $G = K_{r,s}$. Observe o exemplo da Figura 1.9

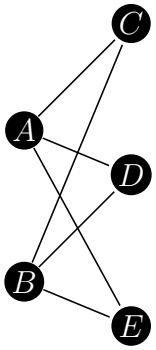

Figura 1.9: Grafo Bipartido $K_{2,3}$.

Definição 1.2.10. Define-se **grafo linha** como o grafo $l(G)$ obtido do grafo G tomando as arestas de G como vértices de $l(G)$ e ligando dois vértices em $l(G)$ quando as arestas correspondentes em G possuírem um vértice em comum. Pode-se perceber. Observe que se G é um grafo k-regular, então $l(G)$ é um grafo $(2k-2)$-regular. Pode-se observar essa propriedade na Figura 1.10:

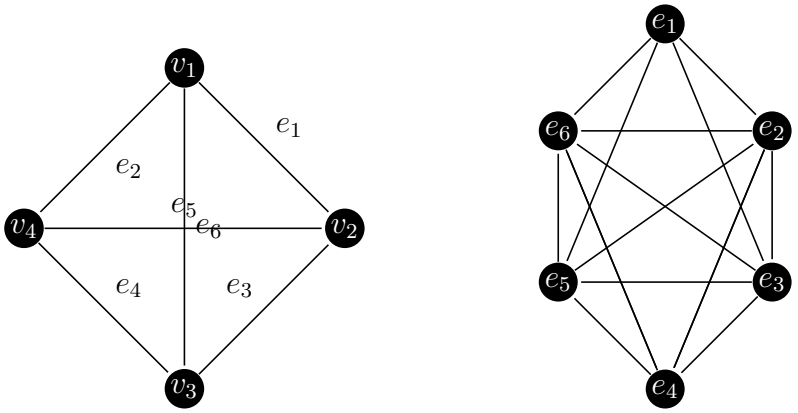

Figura 1.10: K_4 e $l(K_4)$.

Definição 1.2.11. Um **grafo complementar** é o grafo $\overline{G} = \overline{G}(\overline{V}, \overline{E})$ obtido de $G = G(V, E)$ de tal forma que $\overline{V} = V$ e $\{v_i, v_j\} \in \overline{E}$ se e somente se $\{v_i, v_j\} \notin E$ se $i \neq j$. Pode-se observar essa definição na Figura 1.11.

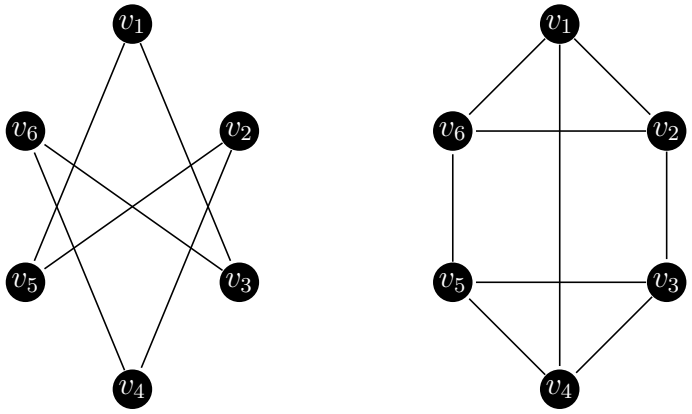

Figura 1.11: Um grafo e seu complementar.

1.3 Relações entre alguns parâmetros de grafos

Definição 1.3.1. Seja $G = G(V, E)$ um grafo. O **grau mínimo** de G é o número:

$$\delta = \min\{d(v) : v \in V\}.$$

O número

$$\Delta = \max\{d(v) : v \in V\}$$

é chamado **grau máximo** de G e

$$\overline{d} = \frac{1}{|V|} \sum_{v \in V} d(v).$$

é chamado de **grau médio** de G.

Essa definição é ilustrada com a Figura 1.12:

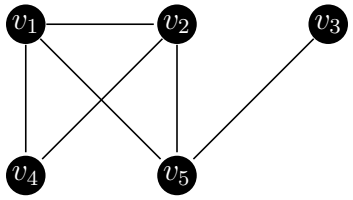

Figura 1.12: $\delta = 1, \Delta = 3, \overline{d} = 2,4$.

Definição 1.3.2. Seja G um grafo conexo. Se v_1 e v_2 são vértices de G, denomina-se de **distância de** v_1 **a** v_2 e denota-se por $d(v_1, v_2)$ ao mínimo dos comprimentos dos caminhos que ligam v_1 a v_2. O máximo das distâncias entre dois vértices quaisquer de G é chamado de **diâmetro** de G e denotado por $diam(G)$. Quando G é um grafo desconexo escreve-se $diam(G) = \infty$.

1.4 Outros Tipos de Grafos

Existem ainda dois tipos de grafos que são importantes mas não serão considerados nesse trabalho: os Dígrafos e os Grafos Ponderados, como será apresentado a seguir.

1.4.1 Grafos Direcionados ou Dígrafos

Anteriormente, estudou-se conceitos de grafos que serão importantes para os métodos que serão apresentados futuramente. Conceitos esses que são sobre grafos não direcionados, ou seja, grafos que as arestas são definidas como pares não ordenados de vértices.

Porém, muitas aplicações práticas as arestas necessitam ser direcionadas dependendo da abordagem que é necessária para o estudo.

Para ilustrar essa situação, será descrito um exemplo para diferenciar grafos não direcionados a dígrafos, também denominados grafos direcionados.

Considere dois grafos (Figura 1.13 e Figuras 1.14), um representa as ruas de um bairro de determinada cidade, e outro, que representa o sentido das ruas dessa mesma cidade. Perceba que apesar de ser a mesma cidade, os grafos podem ser representados de maneiras diferentes, dependendo da abordagem que será aplicada ao problema.

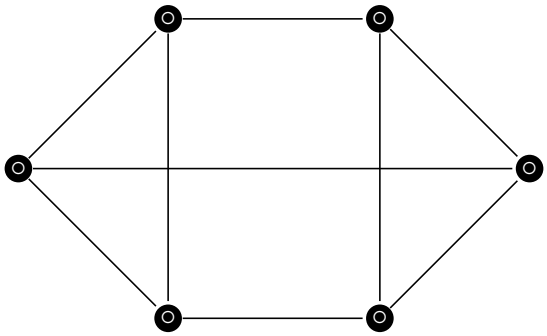

Figura 1.13: Grafo das Ruas de um bairro.

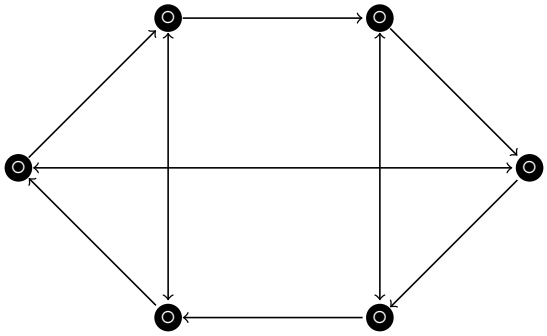

Figura 1.14: Grafo dos Sentidos das Ruas de um bairro.

Alguns exemplos que podem ser utilizados esses tipos de grafos são fluxograma de programas computacionais em que cada vértice representa o comando dado e as arestas a ordem de execução. Dessa forma, atribuí-se sentido às arestas do grafo e possui-se um grafo direcionado ou dígrafo.

1.4.2 Grafos Ponderados

Um grafo é dito ponderado se cada aresta possui determinado peso associado a ela. Suponha que o seguinte grafo representado pela Figura 1.15, denote os possíveis caminhos de uma cidade A à uma cidade G.

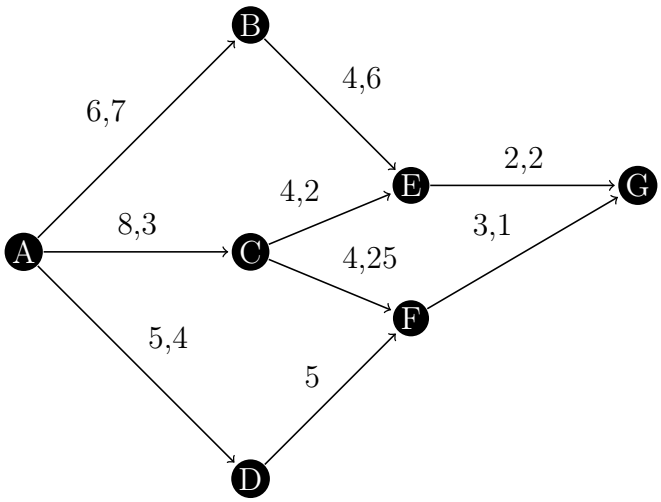

Figura 1.15: Grafo Ponderado

Observe que cada aresta possui um valor associado a ela. Nesse caso, por exemplo, pode representar o custo do pedágio entre duas cidades. Assim, pode ser feita uma análise do melhor caminho a ser percorrido levando em consideração os pesos de cada aresta, mas essa abordagem não será feita no trabalho.

1.5 Matrizes Associadas a um Grafo

Tendo em vista as definições feitas anteriormente, nesta seção, serão caracterizadas algumas propriedades dos grafos, relacionando os resultados para que seja possível desenvolver uma base teórica sólida sobre a Teoria de Grafos.

1.5.1 Matriz de Adjacência

A Matriz de Adjacência tem entradas zeros e uns, e é formada a partir da relação de adjacência entre os vértices. Serão apresentados alguns conceitos sobre essa matriz e técnicas que são utilizadas para deduzir as propriedades estruturais dos grafos. Outra tarefa será relacionar os autovalores da matriz de adjacência e os parâmetros de grafos descritos na seção anterior.

Definição 1.5.1. Seja $G = G(V, E)$ um grafo com n vértices. A **matriz de adjacência** $A(G)$ de G é a matriz quadrada de ordem n cujas entradas A_{ij} são:

$$A_{ij} = \begin{cases} 1 & \text{se } (i,j) \in E \text{ para } v_i, v_j \in V \\ 0 & \text{nos outros casos.} \end{cases}$$

Dessa definição, pode-se concluir que a matriz de adjacência é simétrica para um grafo não direcionado e formada por zeros e uns, como dito anteriormente. Dessa forma, seus autovalores são todos reais. Perceba também que neste caso foi considerado que nenhum vértice é adjacente a ele mesmo, assim todas as entradas da diagonal dessa matriz são zero, e, desta forma o traço da matriz é zero e pode-se concluir que a soma dos seus autovalores é zero.

Definição 1.5.2. O *polinômio característico* da matriz de adjacência $A(G)$ denotado por $p_G(\lambda)$ é definido:

$$\det(\lambda I - A(G)).$$

Define-se, também, λ um **autovalor do grafo G** quando este é raiz do polinômio característico de G, ou seja, $p_G(\lambda) = 0$.

Exemplo 1.5.1. Analisando o espectro do grafo da Figura 1.1 tem-se:

Sua matriz de adjacência é:

$$A(G) = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}.$$

Realizando as contas, obtém-se o polinômio característico:

$$p_G(\lambda) = \lambda^5 - 6\lambda^3 - 4\lambda^2 + 3\lambda + 2.$$

Os autovalores do grafo da Figura 1.1 podem ser calculados pelo software Wolfram:

$$\lambda_1 = 2.6412, \quad \lambda_2 = 0.7237, \quad \lambda_3 = -0.5892, \quad \lambda_4 = -1, \quad \lambda_5 = -1.7757.$$

1.5.2 Matriz Diagonal de Graus

A **matriz de graus** $D(G)$ de G é a matriz diagonal de ordem n cujas entradas D_{ij} são:

$$D_{ij} = \begin{cases} d(v_i) & \text{se } i = j \\ 0 & \text{se } i \neq j. \end{cases}$$

Em outras palavras, a matriz de graus possui as entradas da diagonal principal como sendo o grau de cada vértice.

1.5.3 Matriz Laplaciana

A **Matriz Laplaciana** $L(G)$ é a matriz quadrada de ordem n definida por:

$$L(G) = D(G) - A(G).$$

Essa matriz é essencial no método do corte mínimo que será apresentado futuramente.

1.5.4 Matriz de Incidência

Definição 1.5.3. A matriz de incidência de um grafo G com n vértices e m arestas, denotada por $I(G)$, é a matriz de ordem $n \times m$ cujas entradas são:

$$I_{ij} = \begin{cases} 1 & \text{se } e_j \text{é uma aresta incidente em } v_i V \\ 0 & \text{nos outros casos.} \end{cases}$$

Exemplo 1.5.2. Considere novamente a Figura 1.1 e suas matrizes definidas anteriormente:

$$D = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 3 \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}, \quad L = \begin{bmatrix} 3 & -1 & 0 & -1 & -1 \\ -1 & 3 & -1 & 0 & -1 \\ 0 & -1 & 1 & 0 & 0 \\ -1 & 0 & 0 & 2 & -1 \\ -1 & -1 & 0 & -1 & 3 \end{bmatrix},$$

$$I(G) = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

Capítulo 2

Partições de grafos

Em geral, grafos organizam uma quantidade grande de informações, tornando possível visualizá-las melhor. Contudo, para grafos muito grandes e complexos uma análise visual não é suficiente e faz-se necessário um tratamento analítico do grafo, com o máximo de rigor matemático e o menos empírico possível.

Decompor os grafos em subgrafos, segundo critérios que dependem da análise de interesse, é uma abordagem bastante analítica para os dados de um grafo e mantém a organização e simplicidade de visualização próprias dos grafos.

Existem diferentes abordagens para a partição/decomposição de grafos, cada uma com particularidades e aplicações que dependem das informações que deseja-se extrair. Essa abordagem será feita a partir da análise dos métodos propostos por Newman (2006).

As quatro principais abordagens para particionar um grafo determinam 4 tipos de problema detecção de comunidades: (1) minimização de violações de restrição ou cortes mínimos; (2) maximização da densidade interna ou clusterização; (3) detecção de nós estruturalmente semelhantes no grafo; (4) detecção de blocos que representem a dinâmica do grafo de forma simplificada, por exemplo partições equitativas. Esses métodos podem ser encontrados em Schaub et al(2016).

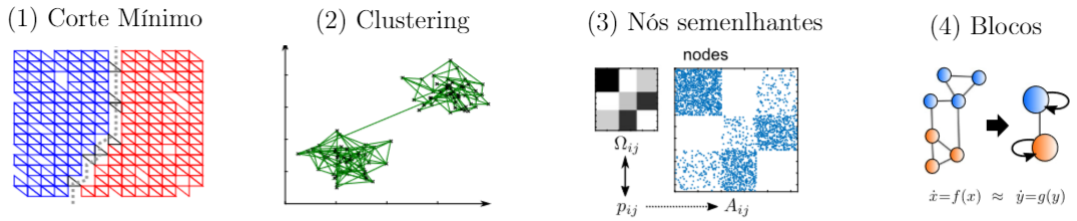

Figura 2.1: Ilustração de quatro métodos de detecção de comunidades. Figura extraída de [8].

Neste trabalho será apresentada a primeira abordagem, Corte Mínimo, e aos métodos mais complexos decorrentes dela.

2.1 Corte Mínimo para a Partição de Grafos

Considere um grafo não-orientado $G = G(V, E)$ e sua matriz de adjacência A. Supondo uma bipartição de G associada a decomposição de V em dois subconjuntos disjuntos V_1 e $V2$, não-nulos. Define-se como corte associado a partição descrita acima o valor

$$R = \frac{1}{2}\sum_{i,j} A_{ij},$$

com $i \in V_1,\ j \in V_2$. Observe que R é o número de arestas de G que conectam vértices que estão em conjuntos diferentes. O fator $\frac{1}{2}$ multiplicando o somatório é para compensar o fato de cada aresta ser contada duas vezes.

Deseja-se que R seja o mínimo. Para isso, R será representada em termos das escolhas dos elementos de V_1 e V_2.

Dessa forma, defina $\vec{s}$ como o vetor de escolha da partição. Esse vetor tem n entradas, sendo n o número de vértices de G e a i-ésima entrada de $\vec{s}$ é

$$s_i = \begin{cases} 1, & \text{se } i \in V_1 \\ -1, & \text{se } i \in V_2. \end{cases}$$

Observe que $\vec{s}^{\,\intercal}\vec{s} = n$.

Note que

$$s_i s_j = \begin{cases} 1, & \text{se } i \text{ e } j \text{ pertencem ao mesmo subconjunto} \\ -1, & \text{se } i \text{ e } j \text{ não pertencem ao mesmo subconjunto.} \end{cases}$$

Consequentemente,

$$\frac{1}{2}(1 - s_i s_j) = \begin{cases} 1, & \text{se } i \text{ e } j \text{ não pertencem ao mesmo subconjunto} \\ 0, & \text{se } i \text{ e } j \text{ pertencem ao mesmo subconjunto.} \end{cases}$$

E assim, pode-se reescrever o corte R em termos do vetor de escolha como

$$R = \frac{1}{2}\sum_{i,j} A_{i,j} = \frac{1}{2}\sum_{i,j}\frac{1}{2}(1 - s_i s_j)A_{ij} = \frac{1}{4}\sum_{i,j}(1 - s_i s_j)A_{ij}.$$

Se k_i é o grau do vértice i então

$$k_i = \sum_j A_{ij}.$$

Observe que

$$\begin{aligned}\sum_{i,j} A_{ij} &= \sum_i k_i \\ &= \sum_i s_i^2 k_i \\ &= \sum_{i,j} s_i s_j \delta_{ij} k_i \\ &= \vec{s}^{\,\intercal} D \vec{s},\end{aligned}$$

em que δ_{ij} é denominado Delta de Kronecker, definido como

$$\delta_{ij} = \begin{cases} 1, & \text{se } i = j \\ 0, & \text{se } i \neq j, \end{cases}$$

a segunda igualdade segue de $s_i^2 = 1$ pois, $s_i = \pm 1$ e a quarta e última igualdade usa notação matricial com D sendo a matriz diagonal de graus.

Também matricialmente, tem-se $\sum_{i,j} s_i s_j A_{ij} = \vec{s}^{\,\intercal} A \vec{s}$. Unindo essas duas igualdades pode-se reescrever R como

$$R = \frac{1}{4} \vec{s}^{\,\intercal} L \vec{s},$$

em que $L = D - A$ é a matriz Laplaciana do grafo. Note que para grafos grandes L possivelmente será esparsa. Ainda, L é simétrica e pode ser obtida do produto da matriz de incidência pela matriz de incidência transposta, $L = G^{\intercal} G$.

Desta forma o tamanho do corte, R, está diretamente associado ao vetor de escolha e o corte mínimo é expresso como o seguinte problema de minimização

$$\min_{\vec{s} \in S} \frac{1}{4} \vec{s}^{\,\intercal} L \vec{s}$$

em que $S \subset \mathbb{R}^n$ e os elementos de S tem entradas ± 1.

A dificuldade deste problema reside em $\vec{s} \in S$, pois $\vec{s}^{\,\intercal} L \vec{s}$ é uma forma quadrática não-negativa.

Diante das boas propriedades de L com respeito ao seu espectro, autovalores reais não-negativos e autovetores, $\vec{v_i}$ que formam uma base ortonormal para o $\mathbb{R}^n$. Serão verificadas agora as possibilidades de $\vec{s}$ em termos dos autovetores de L

Escrevendo $\vec{s}$ como uma combinação linear da base ortonormal de autovetores, $\vec{v_i}$, da matriz Laplaciana, tem-se:

$$\vec{s} = \sum_{i=1}^{n} a_i \vec{v_i},$$

em que $a_i = \vec{s}^{\,\intercal} \vec{v_i}$. Ainda, como $\vec{s}^{\,\intercal} \vec{s} = n$, tem-se:

$$n = \vec{s}^{\,\intercal}\vec{s} = \left(\sum_{i=1}^{n} a_i\vec{v_i}\right)^{\intercal}\vec{s} = \sum_{i=1}^{n} a_i(\vec{v_i}^{\intercal}\vec{s}) = \sum_{i=1}^{n} a_i^2.$$

Então, suprimindo o fator $\frac{1}{4}$, que não interfere na minimização tem-se:

$$R = \left(\sum_{i=1}^{n} a_i\vec{v_i}^{\intercal}\right) L \left(\sum_{j=1}^{n} a_j\vec{v_j}\right) = \sum_{i,j} a_i a_j \lambda_j \delta_{ij} = \sum_{i=1}^{n} a_i^2\lambda_i,$$

em que λ_i é o autovalor associado ao autovetor $\vec{v_i}$.

Sem perda de generalidade, pode-se supor que os autovalores estão indexados em ordem crescente:

$$\lambda_1 \leqslant \lambda_2 \leqslant \lambda_3 \leqslant ... \leqslant \lambda_n.$$

Note que o problema de corte mínimo pode ser formulado como determinar os coeficientes "$a_i's$" tais que $\sum_{i=1}^{n} a_i^2\lambda_i$ seja mínimo e que $\vec{s} \in S$.

Note também que

$$\sum_j L_{ij} = \sum_j (k_i\delta_{ij} - A_{ij}) = k_i - k_i = 0.$$

ou seja, a soma dos termos da i-ésima linha (ou coluna) da matriz Laplaciana é nula. Como consequência $\vec{\mathbf{1}} = (1, 1, \cdots, 1)$ é um autovetor de L associado ao autovalor 0.

Ainda, como já mencionado, todos os autovalores de L são não negativos. Assim, $\lambda_1 = 0$ é o menor autovalor e com autovetor associado $\vec{v_1} = \frac{1}{\sqrt{n}}(1, \cdots, n)$ e consequentemente, de 2.1, $R \geq 0$.

Portanto, minimizar R corresponde a escolher $a_1 = 1, a_2 = 0, \cdots, a_n = 0$, ou seja, escolher $\vec{s} = (1, \cdots, 1)$.

Essa solução, $\vec{s} = \vec{\mathbf{1}}$, representa a partição trivial, com todos os vértices em um único grupo e nenhum vértice no outro grupo.

Existem alguns meios de evitar essa solução trivial: Fixar o tamanho dos dois grupos. Se um grupo de tamanho n_1 e o outro n_2, então note que a_1 será fixado pois,

$$a_1^2 = (\vec{v_1}^{\intercal}\vec{s}) = \frac{(n_1 - n_2)}{2}.$$

mas na prática, a_1 não afeta a minimização de R, pois está associado ao autovalor 0.

Com essa restrição, em a_1 e a restrição $\vec{s}^{\intercal}\vec{s} = n$ para o vetor de escolha, a minimização de R ocorreria, idealmente, se fosse escolhido $\vec{s}$ proporcional a $\vec{v_2}$, o autovetor associado ao segundo menor autovalor. $\vec{v_2}$ é chamado de vetor de Fiedler. O segundo menor autovalor λ_2, também é conhecido como *conectividade algébrica*. Porém, existe uma restrição adicional de que $\vec{s} \in S$, ou seja, $s_i = \pm 1$ o que, em geral, impede que $\vec{s}$ seja paralelo a $\vec{v_2}$.

Como já comentado, a restrição $\vec{s} \in S$ dificulta bastante o problema de minimização. Uma alternativa é escolher $\vec{s}$ o mais próximo possível de ser paralelo a $\vec{v_2}$, ou equivalentemente, o mais distante possível de ser ortogonal $\vec{v_2}$.

Traduzindo essa escolha, obtem-se:

$$\max_{\vec{s}} |\vec{v_2}^{\intercal}\vec{s}| = \max_{\vec{s}} \left| \sum_i v_{2_i} s_i \right|.$$

Observe que :

$$\max_{\vec{s}} \left| \sum_{i=1}^{n} v_{2_i} s_i \right| \leqslant \sum_i |v_{2_i}|.$$

O máximo de $|\sum_{i=1}^{n} v_{2_i} s_i|$ será $\sum_i |v_{2_i}|$ se v_{2_i} e s_i tiverem o mesmo sinal para todo $i \in \{1, \cdots, n\}$, ou seja, a solução é $\vec{s}$ tal que

$$s_i = \begin{cases} 1, & \text{se } v_{2_i} > 0 \\ -1, & \text{se } v_{2_i} < 0, \end{cases}$$

resta ainda decidir o valor de s_i no caso $v_{2_i} = 0$.

Contudo essa solução/aproximação ainda não impõe que os grupos tenham n_1 e n_2 vértices, ou seja, não impõe que $\vec{s}$ tenha n_1 entradas com valor 1 e n_2 entradas com valor -1.

Duas possibilidades para satisfazer essa restrição e fazer s aproximadamente paralelo a $\vec{v_2}$ são:

- $s_i = 1$ para os n_1 maiores valores de v_{2_i} e $s_i = -1$ para os demais.
- $s_i = -1$ para os n_2 menores valores de v_{2_i} e $s_i = 1$ para os demais.

Observe que se $\lambda_2 = 0$ o grafo é desconexo, ou seja, está naturalmente bipartido em dois grupos não vazios. Com isso, para que o método da partição apresentado funcione bem, precisa-se que λ_2 suficientemente grande. Entretanto, isso não será explorado nessa parte.

2.2 Estrutura de Comunidade e Modularidade

Uma propriedade que se espera de uma comunidade em uma determinada rede é que os elementos de cada comunidade (ou os vértices de cada partição) estejam mais conectados que o esperado.

Nosso objetivo agora é decompor o grafo em comunidades com essas características, ou seja, encontrar grupos em que a quantidade de arestas dentro de cada grupo é maior que o esperado.

Para isso, utiliza-se uma função denominada *modularidade*, que é definida como:

$$Q = (\text{número de arestas dentro da comunidade}) - (\text{número esperado de tais arestas}).$$

Espera-se que número de arestas dentro de uma comunidade seja mais facilmente encontrado. Assim, o foco do problema está na verificação do número esperado de arestas dentro dessa comunidade.

Portanto, considere um grafo $G(V, E)$, tal que P_{ij} é definida como a probabilidade dos vértices $i, j \in V$ estarem conectados.

Sendo m o número de arestas de G, a função modularidade pode ser escrita como:

$$Q = \frac{1}{2m} \sum_{i,j} [A_{ij} - P_{ij}] \delta_{g_i, g_j}.$$

com

$$\delta_{g_i, g_j} = \begin{cases} 1, \text{ se i e j estão nos mesmos grupos} \\ 0, \text{caso contrário.} \end{cases}$$

Note que quando escreve-se a modularidade dessa maneira, está se somando todos os vértices que estão numa mesma comunidade. Isso é determinado pelo δ_{g_i, g_j}, que irá somar 0 caso dois vértices estejam em grupos separados.

Agora, sobre escolha de P_{ij} precisa-se considerar grafos não direcionados, ou seja, $P_{ij} = P_{ji}$, e também que $Q = 0$ possui o significado de que todos os vértices estão em uma mesma comunidade.

Existem algumas possibilidades de modularidade nula, mas apenas uma é compatível com o que se espera na realidade. Para satisfazer essa demanda, serão considerados modelos nos quais o grau esperado em cada vértice do grafo modelo é igual ao real grau do vértice correspondente na comunidade real. Essa condição é

$$\sum_j P_{ij} = k_i.$$

Se essa condição é satisfeita, então

$$2m = \sum_{i,j} P_{ij} = \sum_{i,j} A_{ij} = \sum_i k_i. \tag{2.1}$$

Um modelo possível, consiste em escolher P_{ij} randomicamente, estando sujeito apenas a restrição acima. Observe que essa escolha resulta em uma modularidade diferente a cada aplicação do método.

Outra alternativa passa por observar que, a probabilidade de que uma aresta esteja ligada ao vértice i depende principalmente do grau k_i do vértice, e a probabilidade de dois vértices estarem ligados a uma aresta são independentes uma da outra. Isso implica que o número esperado de arestas P_{ij} entre os vértices i e j é bem representado por um produto $f(k_i)f(k_j)$ de funções que dependem dos respectivos graus, mas a função f deve ser a mesma, para que esteja de acordo com o fato de P ser simétrica, $(P_{ij} = P_{ji})$.

Dessa forma, tem-se que

$$\sum_{j=1}^{m} P_{ij} = f(k_i) \sum_{j=1}^{m} f(k_j) = k_i c \quad \forall i,$$

e, consequentemente,

$$f(k_i) = ck_i, \tag{2.2}$$

com c uma constante.

Unindo as Equações 2.2 e 2.1 , tem-se que:

$$2m = \sum_{i,j} P_{ij} = c^2 \sum_{i,j} k_i k_j = (2mc)^2 \Rightarrow c = \frac{1}{\sqrt{2m}}.$$

e

$$P_{ij} = \frac{k_i k_j}{2m}.$$

Assim, para este modelo, a modularidade é definida como:

$$Q = \frac{1}{2m} \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta_{g_i, g_j}. \tag{2.3}$$

2.3 Otimização Espectral da Modularidade

Tendo em vista a construção de uma expressão explícita para a modularidade, o objetivo agora é maximizá-la, considerando todas as possíveis divisões do grafo, ou seja, abordar a modularidade de maneira matricial, podendo tirar conclusões a partir da análise espectral da função modularidade.

2.3.1 Método dos autovetores

Nesse momento, será considerado o problema de particionar o grafo em duas comunidades. Igualmente ao que foi proposto na Seção 2.2. Considere o vetor de escolha $\vec{s}$ tal que:

$$s_i = \begin{cases} 1, & \text{se } i \in V_1 \\ -1, & \text{se } i \in V_2. \end{cases}$$

Note, novamente, que

$$\frac{1}{2}(s_i s_j + 1) = \begin{cases} 1, \text{ se i e j pertencem ao mesmo grupo} \\ 0, \text{ se i e j pertencem a grupos diferentes,} \end{cases}$$

ou seja,

$$\delta(g_i, g_j) = \frac{1}{2}(s_i s_j + 1).$$

Assim, pode-se reescrever (2.3) da seguinte maneira:

$$\begin{aligned} Q &= \frac{1}{2m}\sum_{i,j}[A_{ij} - P_{ij}]\delta_{g_i,g_j} \\ &= \frac{1}{4m}\sum_{i,j}[A_{ij} - P_{ij}](s_i s_j + 1). \end{aligned}$$

Considerando ainda (2.1), tem-se

$$\begin{aligned} Q &= \frac{1}{4m}\sum_{i,j}[A_{ij} - P_{ij}](s_i s_j + 1) \\ &= \frac{1}{4m}\left(\sum_{ij}[A_{ij} - P_{ij}]s_i s_j + \sum_{ij} A_{ij} - P_{ij}\right) \\ &\overset{(2.1)}{=} \frac{1}{4m}\sum_{i,j}[A_{ij} - P_{ij}]s_i s_j, \end{aligned}$$

e matricialmente,

$$Q = \frac{1}{4}\vec{s}^{\,\intercal} B\vec{s} \qquad , \tag{2.4}$$

a qual B é uma matriz simétrica real, com elementos tais que $B_{ij} = A_{ij} - P_{ij}$, chamada matriz de modularidade. B terá um papel semelhante ao da matriz Laplaciana no método do corte mínimo.

Observe que

$$\sum_j B_{ij} = \sum_j A_{ij} - \sum_j P_{ij} = k_i - k_i = 0,$$

e consequentemente, $\vec{\mathbf{1}}$ é o autovetor associado ao autovalor 0, analogamente a matriz Laplaciana. Porém, ao contrário da Laplaciana, os autovalores da matriz de modularidade não são todos não negativos. Em geral, existem autovalores positivos e negativos.

Partindo de (2.4), e escrevendo $\vec{s}$ como uma combinação linear dos autovetores normalizados $\vec{u}_i$.

$$\vec{s} = \sum_{i=1}^{n} a_i \vec{u}_i, \tag{2.5}$$

com $a_i = \vec{u_i}^{\intercal}\vec{s}$

Assim, reescreve-se a função modularidade como

$$Q = \frac{1}{4m}\sum_i a_i^2 \beta_i.$$

em que β_i é uma autovalor de B associado ao autovetor $\vec{u_i}$.

Agora, deve-se assumir, sem perda de generalidade, que os autovalores estão indexados em ordem decrescente, isto é, $\beta_1 \geqslant \beta_2 \geqslant \cdots \geqslant \beta_m$, e a tarefa de maximizar Q passa a ser escolher a_i de modo que Q maximizada, isto é, colocando o "maior peso" possível no maior autovalor

(positivo).

Se não houver restrições para $\vec{s}$, bastaria escolhe-lo proporcional a $\vec{u_1}$. Mas $s_i = \pm 1$, o que, em geral, impede que $\vec{s}$ seja escolhido paralelo a $\vec{u_1}$.

Usando o mesmo método da partição espectral, uma boa aproximação é escolher $\vec{s}$ o mais próximo possível de ser paralelo a $\vec{u_1}$, isto é, escolhendo $\vec{s}$ como

$$s_i = \begin{cases} +1, \text{ se } \vec{u_{1,i}} \geqslant 0 \\ -1, \text{ se } \vec{u_{1,i}} < 0. \end{cases}$$

2.3.2 Outros autovetores da matriz de modularidade

O algoritmo da sessão anterior possui duas limitações/deficiências: Divide o grafo em apenas duas comunidades, o que, em geral, não representa a realidade e utiliza apenas o autovetor associado ao maior autovalor, ignorando todos os outros.

Para remediar essa deficiência, a generalização do método espectral, será apresentada a proposta de Alpert e Yao [1] e por White e Smythy [7].

Defina S uma matriz de índices $n \times c$, em que n é o número de vértices e c o número de comunidades existentes. Essa matriz tem cada coluna como sendo o vetor de escolha de uma comunidade $\mathbf{S} = (s_1|s_2|\cdots|s_c)$ tal que:

$$s_i = \begin{cases} 1, \text{ se o vértice } i \text{ pertence a comunidade } j \\ 0, \text{ caso contrário} \end{cases} .$$

Note que $\vec{s_i} \perp \vec{s_j}\ \forall\ i \neq j$, além disso, $Tr(S^\intercal S) = n$.

Agora, redefini-se $\delta(g_i, g_j)$ como :

$$\delta(g_i, g_j) = \sum_{k=1}^{c} (S_{ik} S_{jk}), \tag{2.6}$$

e suprimindo o fator $\dfrac{1}{2m}$ a função modularidade para essa divisão é

$$Q = \sum_{i,j=1}^{n} \sum_{k=1}^{c} B_{ij} S_{ik} S_{jk} = Tr(S^\intercal B S), \tag{2.7}$$

em que B é a matriz de modularidade definida anteriormente. Assim, escrevendo $B = UDU^\intercal$, onde $U = (u_1|u_2|\cdots)$ é a matriz dos autovetores de B e D é a matriz diagonal de autovalores $D_{ii} = \beta_i$, e assim:

$$Q = \sum_{j=1}^{n} \sum_{k=1}^{c} \beta_j (u_j^\intercal s_k)^2.$$

Novamente, o objetivo é maximizar a modularidade, mas agora sem restrições ao número de comunidades, ou seja, S pode ter tantas colunas quanto precisa-se para maximizar Q.

Se os elementos de Q não tivessem restrições, uma escolha de c comunidades é equivalente a

escolher $c-1$ colunas independentes e ortogonais $\vec{s}_1, \cdots, \vec{s}_c$. Neste caso, possui-se um caminho claro: Q seria maximizada escolhendo-se as colunas proporcionais aos principais autovetores de B.

Entretanto, apenas autovetores associados a autovalores positivos contribuem, de fato, para a maximização da modularidade. Dessa maneira, a modularidade ótima seria obtida escolhendo exatamente tantas colunas para S quantos os autovalores positivos de B, ou, equivalentemente, escolher o número de grupos c como um número a mais do que número de autovalores positivos.

Entretanto, existe a restrição de que $\vec{s}_i$ é um vetor composto por 0 e 1, ou seja, é possível não encontrarmos tantos vetores de índice que contribuam positivamente para a maximização da modularidade em relação com os autovalores positivos de B.

Assim, o número de autovalores positivos mais um é um limitante superior para número de comunidades.Novamente, percebe-se que há uma conexão entre a matriz de modularidade e a estrutura de comunidades da rede que ela descreve.

2.3.3 Algoritmo de Partição Vetorial

Generalizando a maximização da modularidade, deve-se considerar p autovalores principais ao contrário dos métodos anteriores que consideravam apenas um e, nesse caso, p pode ser qualquer número entre 1 e n. Entretanto, alguns dos autovalores podem ser negativos, o que é inconveniente para a maximização.

Observação 2.3.1. Note que

$$[U^\intercal S]_{jk} = \vec{u}_j^\intercal \vec{s}_k = \sum_{j=1}^{n} U_{ij} S_{ik}.$$

Ainda

$$Tr(XY^\intercal) = \sum_{i,j} X_{ij} Y_{ij}.$$

Assim

$$Tr((U^\intercal S)(U^\intercal S)^\intercal) = \sum_{i,j} (U^\intercal S)_{ij}^2. \tag{2.8}$$

Partindo de (2.7):

$$\begin{aligned} Q &= Tr(S^\intercal BS) \qquad (2.9)\\ &= Tr(S^\intercal UDU^\intercal S) \quad (\text{ escrevendo } B = UDU^\intercal)\\ &= Tr(S^\intercal U(D - \alpha I + \alpha I)U^\intercal S)\\ &= Tr(S^\intercal U(D - \alpha I)U^\intercal S + S^\intercal U\alpha IU^\intercal S)\\ &= Tr(S^\intercal U(D - \alpha I)U^\intercal S) + Tr(S^\intercal U\alpha IU^\intercal S)\\ &= Tr(S^\intercal U(D - \alpha I)U^\intercal S) + \alpha Tr(S^\intercal UU^\intercal S)\\ &= Tr(S^\intercal U(D - \alpha I)U^\intercal S) + \alpha Tr(S^\intercal S)\\ &= n\alpha + Tr(S^\intercal U(D - \alpha I)U^\intercal S). \qquad (2.10) \end{aligned}$$

Note que

$$\begin{aligned} & Tr(S^\intercal_{c\times n}U_{n\times n}(D - \alpha I)_{n\times n}U^\intercal_{n\times n}S_{c\times n})\\ &= Tr(U^\intercal_{n\times n}S_{n\times c}S^\intercal_{c\times n}U_{n\times n}(D - \alpha I))\\ &= \sum_{j=1}^{n}(\beta_j - \alpha)[U^\intercal SS^\intercal U]_{jj}. \qquad (2.11) \end{aligned}$$

Denominando $U^\intercal S = M^\intercal_{n\times c}$ e $S^\intercal U = M_{c\times n}$ tem-se,

$$\begin{aligned} \sum_{j=1}^{n}(\beta_j - \alpha)[M^\intercal_{n\times c}M_{n\times c}]_{jj} &= \sum_{j=1}^{n}(\beta_j - \alpha)\left[\sum_{k=1}^{c}(M_{jk}M_{jk})\right]\\ &= \sum_{j=1}^{n}(\beta_j - \alpha)\sum_{k=1}^{c}(M_{jk})^2. \qquad (2.12) \end{aligned}$$

Perceba que

$$M_jk = [U^\intercal S]_{jk} = \vec{u}^\intercal_j\vec{s}_k = \sum_{i=1}^{n}U_{ij}S_{ik},$$

então, retornando a Equação (2.12),

$$\begin{aligned} \sum_{j=1}^{n}(\beta_j - \alpha)\sum_{k=1}^{c}(M_{jk})^2 &= \sum_{j=1}^{n}(\beta_j - \alpha)\sum_{k=1}^{c}\left(\sum_{i=1}U_{ij}S_{ik}\right)\\ &= \sum_{j=1}^{n}\sum_{i=1}^{c}(\beta_j - \alpha)\left(\sum_{i=1}^{n}U_{ij}S_{ik}\right). \qquad (2.13) \end{aligned}$$

Portanto, da Equação (2.10):

$$\begin{aligned} Q &= n\alpha + \sum_{j=1}^{n}\sum_{k=1}^{c}\left[(\beta_j - \alpha)\left(\sum_{i=1}^{n} U_{ij}S_{ik}\right)^2\right] \\ &= n\alpha + \sum_{j=1}^{p}\sum_{k=1}^{c}\left[\sum_{i=1}^{n}\sqrt{(\beta_j - \alpha)}U_{ij}S_{ik}\right]^2. \\ &= n\alpha + \sum_{j=1}^{n}\sum_{k=1}^{c}(\beta_j - \alpha)\left[\sum_{i=1}^{n} U_{ij}S_{ik}\right]^2. \end{aligned} \tag{2.14}$$

Defini-se um conjunto de vetores associados aos vértices $\vec{r_i}$, com $i = 1, \cdots, n$ de dimensão p, ou seja, p é da forma $p \times 1$ e são definidos como

$$[r_i]_j = \sqrt{\beta_j - \alpha}\; U_{ij}, \tag{2.15}$$

contanto que $\alpha \leqslant \beta_p$, $\vec{r_i}$ será real para todo $i = 1, \cdots, n$.

Então,

$$\begin{aligned} Q &\simeq n\alpha + \sum_{j=1}^{p}\sum_{k=1}^{c}\left[\sum_{i=1}^{n}\sqrt{\beta_j - \alpha}\; U_{ij}S_{ik}\right]^2 \\ &= n\alpha + \sum_{k=1}^{c}\sum_{j=1}^{p}\left[\sum_{i\in G_k}[r_i]_j\right] \qquad (2.16) \\ &= n\alpha + \sum_{k=1}^{c}|R_k|^2, \qquad (2.17) \end{aligned}$$

onde G_k é o conjunto dos vértices do grupo k e os *vetores de comunidade* R_k $k = 1, \cdots, c$ são

$$R_k = \sum_{i\in G_k} r_i.$$

Portanto, o problema de encontrar estruturas de comunidades é equivalente a escolher uma divisão dos vértices em grupos de modo que a norma dos vetores $\vec{R}_k$ seja maximizada.

Isso significa que é necessário agrupar os vetores de vértices individuais $\vec{r_i}$ que tenham aproximadamente a mesma direção e sentido.

Já foi observado que o parâmetro "p" corresponde a quantidade de autovalores e autovetores que vão contribuir para a resolução do problema e controla o balanço entre a complexidade do problema, bem como a precisão da aproximação feita. A escolha de "p"é definida conforme o problema, mas não se entrará nesse assunto.

Caso p seja escolhido de forma que seja pequeno existem algumas consequências: uma menor complexidade do problema, um menor custo computacional e menor precisão da aproximação.

Porém, se $n = p$ então a Equação 2.14 se iguala a Equação 2.16 e assim,

$$\vec{r}_i^{\intercal}\vec{r}_j \overset{(\text{def})}{=} \sum_{k=1}^{n} U_{ik}(\beta_k - \alpha)U_{jk} \overset{(\text{def})}{=} B_{ij} - \alpha\delta_{ij}.$$

Ou seja (2.16) traduz o problema de maximizar a modularidade em um problema de partição vetorial. Entretanto, na prática, a vantagem ocorre quando escolhe-se $p < n$, em que o método é responsável por extrair os fatores que mais contribuem para maximizar a modularidade. Em outras palavras, escolhe os maiores autovetores.

Será considerado o caso da partição ser feita em duas comunidades. Como $\vec{\mathbf{1}}$ é um autovetor da matriz de modularidade e os autovetores dela são ortogonais pode-se concluir que

$$\sum_{i=1}^{n}[\vec{u}_j]_i = \sum_{i=1}^{n} 1.[\vec{u}_j]_i = \sqrt{n}\ \vec{\mathbf{1}}^{\intercal}\ \vec{u}_j = \sqrt{n}\ \vec{u}_1^{\intercal}\ \vec{u}_j = 0. \tag{2.18}$$

Por definição (2.15) sabe-se que:

$$\sum_{i=1}^{n}[\vec{r}_i]_j = \sqrt{\beta_j - \alpha}\sum_{i=1}^{n} U_{ij} = \sqrt{\beta_j - \alpha}\sum_{i=1}^{n}[\vec{u}_j]_i \overset{(2.18)}{=} 0,$$

e portanto

$$\sum_{i=1}^{n}\vec{r}_i = 0, \tag{2.19}$$

para todo p. Isso implica que os vetores de comunidades $\vec{R}_k$ também somam zero, isto é,

$$\sum_{k=1}^{c}\vec{R}_k = \sum_{k=1}^{c}\sum_{i\in G_k}\vec{r}_i \overset{(2.19)}{=} \vec{0}.$$

Em particular, se o objetivo é particionar o grafo em duas comunidades, obtem-se somente os vetores de comunidade $\vec{R}_1$ e $\vec{R}_2$, que são iguais em magnitude e em direções opostas.

Além disso, o máximo da modularidade (2.16) é alcançado quando o vetor de vértice $\vec{r}_i$, individualmente, possui produto interno positivo com o vetor comunidade ao qual o vértice pertence.

Ilustra-se isso considerando a remoção de $\vec{r}_i$ de $\vec{R}_k$ se $\vec{R}_k \cdot \vec{r}_i < 0$ e pode-se analisar a mudança com relação a $|\vec{R}_k|^2$:

$$|\vec{R}_k - \vec{r}_i|^2 - |\vec{R}_k|^2 = |\vec{r}_i|^2 - 2.\vec{R}_k.\vec{r}_i > 0. \tag{2.20}$$

Basta verificar que

$$\begin{aligned}|\vec{R}_k - \vec{r}_i|^2 &= \langle\vec{R}_k - \vec{r}_i, \vec{R}_k - \vec{r}_i\rangle = \langle\vec{R}_k, \vec{R}_k\rangle - 2\langle\vec{R}_k, \vec{r}_i\rangle + \langle\vec{r}_i, \vec{r}_i\rangle \\ &= |\vec{R}_k|^2 - 2\langle\vec{R}_k, \vec{r}_i\rangle + |\vec{r}_i|^2,\end{aligned}$$

e subtraindo $|\vec{R}_k|^2$ dos dois lados da igualdade, obtem-se a igualdade 2.20.

Analogamente, adicionando um vértice i para o qual $\vec{R}_k \cdot \vec{r}_i > 0$ o valor de $|\vec{R}_k|^2$ é aumentado.

Considerando essas propriedades a partição ótima de uma rede em duas é determinada por uma direção ótima para o vetor $\vec{R}_1$.

Sabendo qual é o vetor $\vec{R}_1$, seria possível determinar o plano perperdicular a $\vec{R}_1$ e que passa pela origem do espaço vetorial p-dimensional e, consequentemente, seria possível determinar de que lado do plano está $\vec{r}_i$.

2.3.4 A Escolha de α

Na Equação (2.15) observa-se a existência de uma constante α. Essa constante está relacionada com o valor de p e afeta a precisão na aproximação de Q. Ao desconsiderar os $n-p$ autovalores mais negativos, aproxima-se $B - \alpha I = U(D - \alpha I)U^\intercal$ por $U(D' - \alpha I')U^\intercal$, em que D' e I' são iguais a D e I a menos das últimas $n-p$ entradas de suas diagonais.

Agora, quantifica-se o erro dessa aproximação:

$$\chi^2 = Tr[U(D - \alpha I)U^\intercal - U(D' - \alpha I')U^\intercal]^2 \tag{2.21}$$

$$= Tr[(D - \alpha I) - (D' - \alpha I')]^2 = \sum_{i=p+1}^{n} (\beta_i - \alpha)^2. \tag{2.22}$$

Minimiza-se o erro considerando que $\dfrac{d\chi^2}{d\alpha} = 0$, ou seja, quando

$$\begin{aligned}
\frac{d\chi^2}{d\alpha} = 0 &\Leftrightarrow \sum_{i=p+1}^{n} -2(\beta_i - \alpha) = 0 \\
&\Leftrightarrow 2(n-p)\alpha - 2\sum_{i=p+1}^{n} \beta_i = 0 \\
&\Leftrightarrow \alpha = \frac{1}{n-p}\sum_{i=p+1}^{n} \beta_i.
\end{aligned} \tag{2.23}$$

Ou seja, o erro quadrático médio mínimo introduzido por nossa aproximação é obtido ajustando α igual à média dos autovalores que foram descartados.

Embora seja uma forma diferente e interessante de analisar o problema de detecção de comunidades, a partição vetorial tem a mesma complexidade. Não será considerada essa estratégia nos experimentos.

Capítulo 3

Experimentos

Foi implementado um algoritmo no software R, inserido no Apêndice, com o objetivo aplicar os métodos estudados anteriormente e corroborar os resultados apresentados em [5].

Como o desenvolvimento teórico considerou a divisão em duas comunidades, aplicar os métodos em grafos onde sabe-se *a priori* que houve uma divisão é uma forma de validar os métodos. No primeiro experimento essa característica é conhecida. No segundo experimento o grafo em questão não representa uma rede que foi dividida em duas.

3.1 O Problema dos Golfinhos

Um *benchmark* com a característica acima descrita é o problema da comunidade de Golfinhos cujo comportamento foi monitorado e apresentado em [2] e os dados estão disponíveis em [6]. Nesse grupo de golfinhos foi constatado uma separação em dois grupos após determinada situação.

As Figuras 3.1, 3.2 e 3.3 apresentam as comunidades detectadas pelos métodos da Modularidade, Corte Mínimo Livre e Corte mínimo, respectivamente, realizadas no *software* R, e teve um processamento rápido dos dados. Corte Mínimo Livre é como se denomina aplicação do método do corte mínimo sem restrição a quantidade de elementos em cada comunidade.

No grafo, os vértices em forma de quadrado e círculo representam as duas comunidades após a separação natural dos golfinhos. Para distinguir as comunidades detectadas pelos métodos apresentados colore-se os vértices em duas cores diferentes, azul e laranja. Não se sabe ao certo qual o significado de cada aresta no grafo.

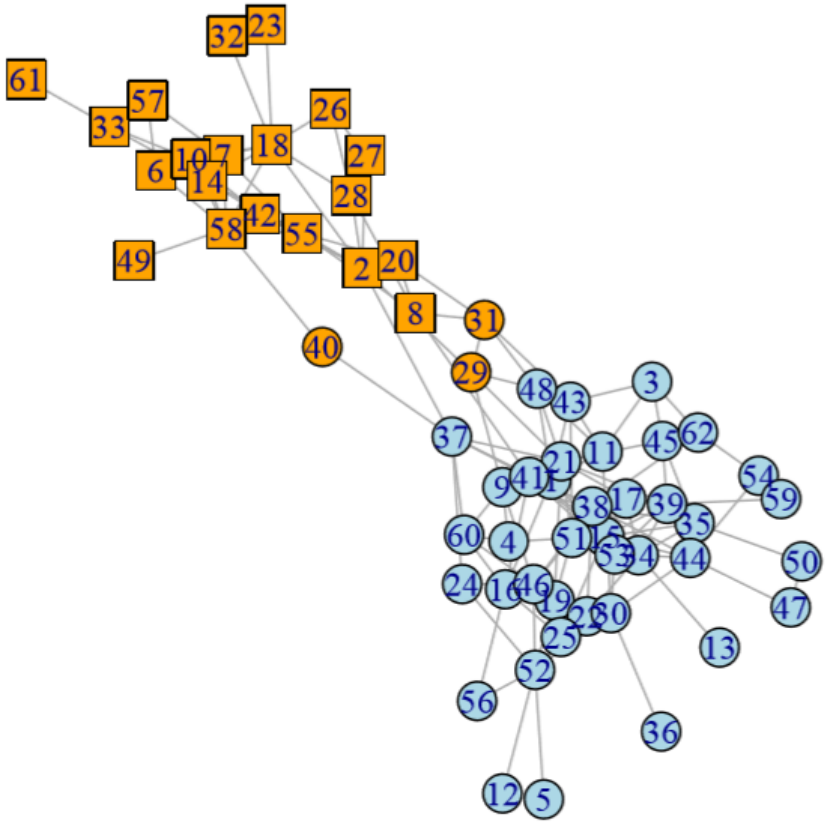

Figura 3.1: Detecção de Comunidades pelo método da Modularidade.

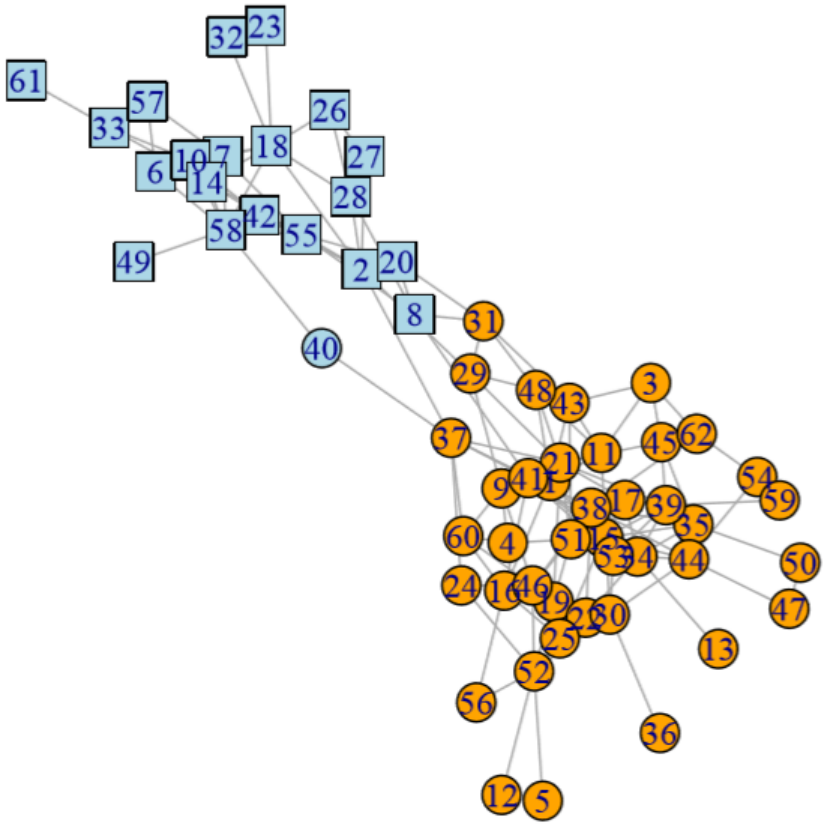

Figura 3.2: Detecção de Comunidades pelo método do Corte Livre.

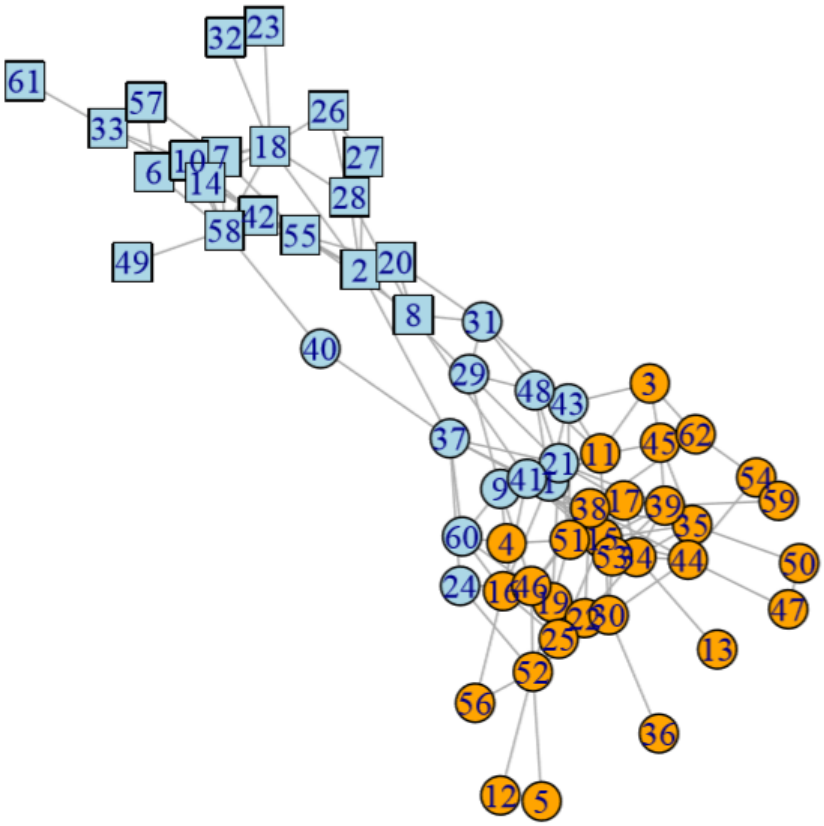

Figura 3.3: Detecção de Comunidades pelo método do Corte Mínimo, com comunidades de mesma ordem.

No caso da Figura 3.3 foi determinado no algoritmo que a partição seria feita de forma que uma comunidade tenha 32 elementos, já que ao todo são 64 vértices, que representam os golfinhos.

Note que os dois primeiros métodos tiveram uma melhor aproximação da realidade. Isso é perceptível observando a quantidade de vértices em forma de círculo que estão na comunidade laranja, mas que na separação natural estão no outro grupo. No caso da Modularidade, isso acontece com três golfinhos apenas e no Corte Livre apenas um. Já no método do Corte que divide o grafo em duas comunidades com quantidades iguais de elementos esse número foi de doze golfinhos.

3.2 O Grafo das Linhas Aéreas Norte Americanas

Neste experimento escolhe-se uma rede que representa um conjunto de linhas aéreas Norte Americanas de 1997. Esses dados também podem ser encontrados no repositório *Network Repository* (*http://networkrepository.com/*).

Nesse problema, não existe uma separação natural do grafo em duas comunidades, como no caso anterior.

O código do qual os resultados desse experimento foram obtidos também está no Apêndice.

Junto com os dados citados acima está um outro *link* que possui a forma que o grafo original possui e será apresentado na Figura 3.4:

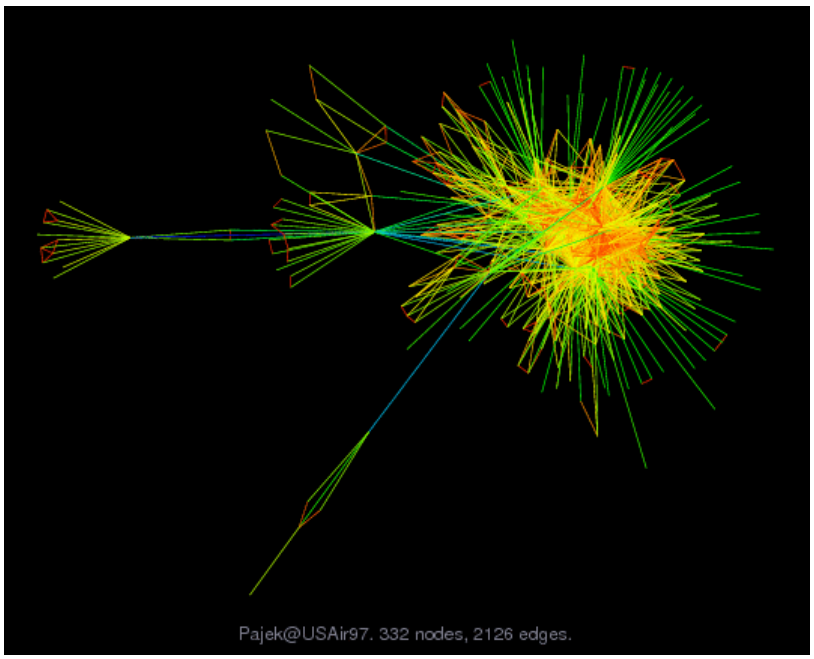

Figura 3.4: Imagem retirada de *https://www.cise.ufl.edu/research/sparse/matrices/Pajek/USAir97*.

No algoritmo as comunidades identificadas estão separadas em azul e laranja novamente. Perceba a semelhança existente entre a Figura 3.4 e as Figuras 3.5, 3.6 e 3.7, lembrando que essas foram feitas a partir do algoritmo.

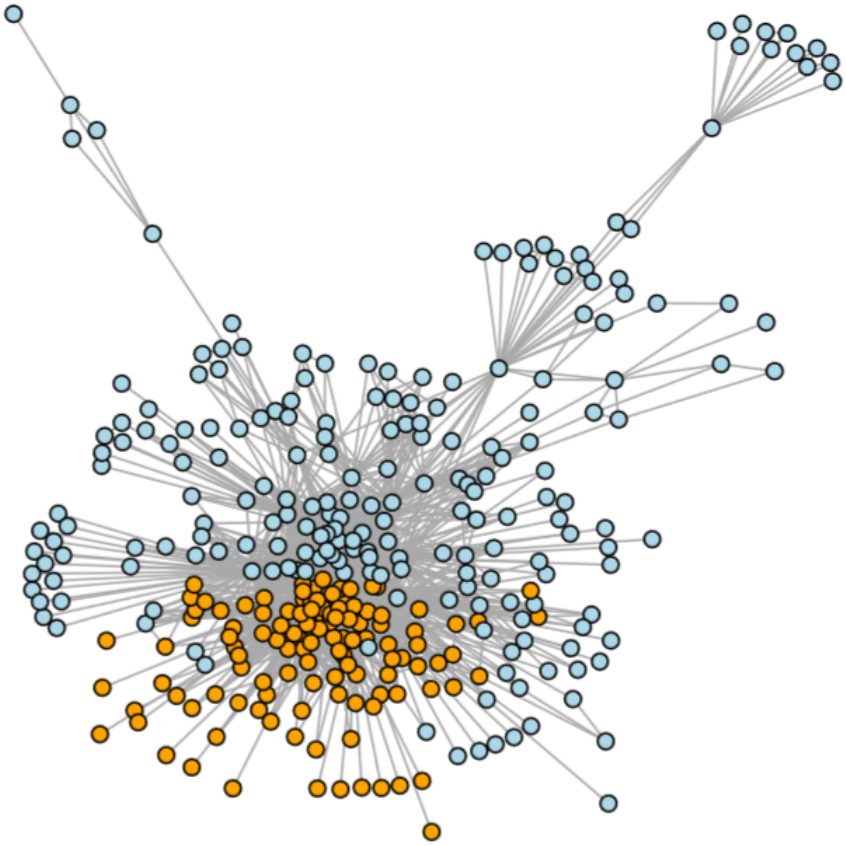

Figura 3.5: Detecção de Comunidades pelo método da Modularidade.

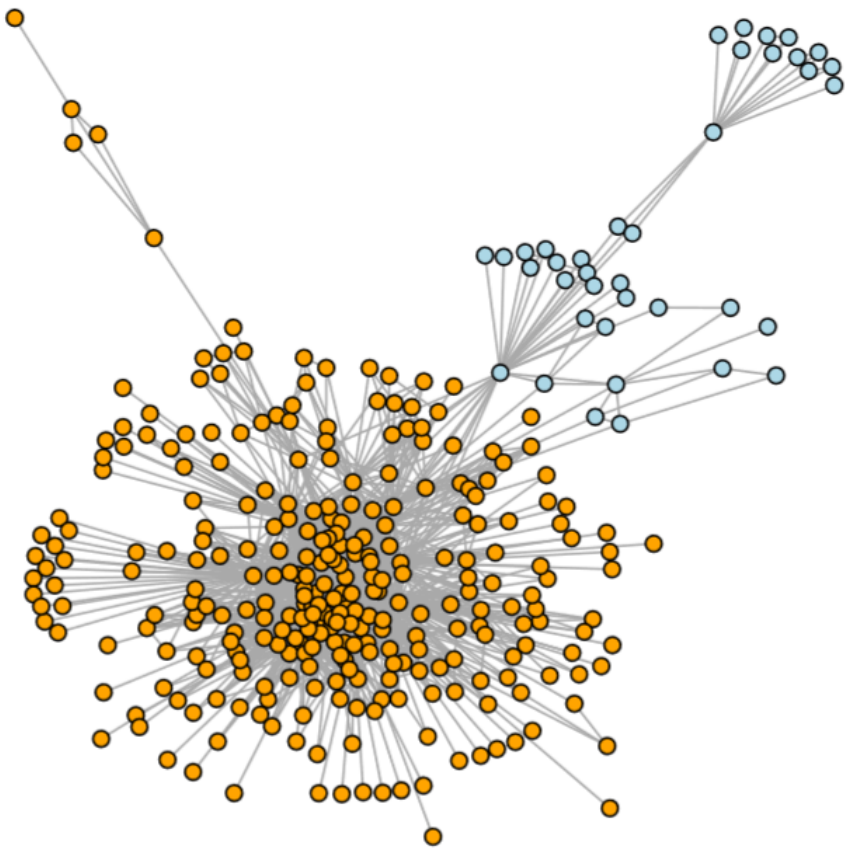

Figura 3.6: Detecção de Comunidades pelo método do Corte Livre.

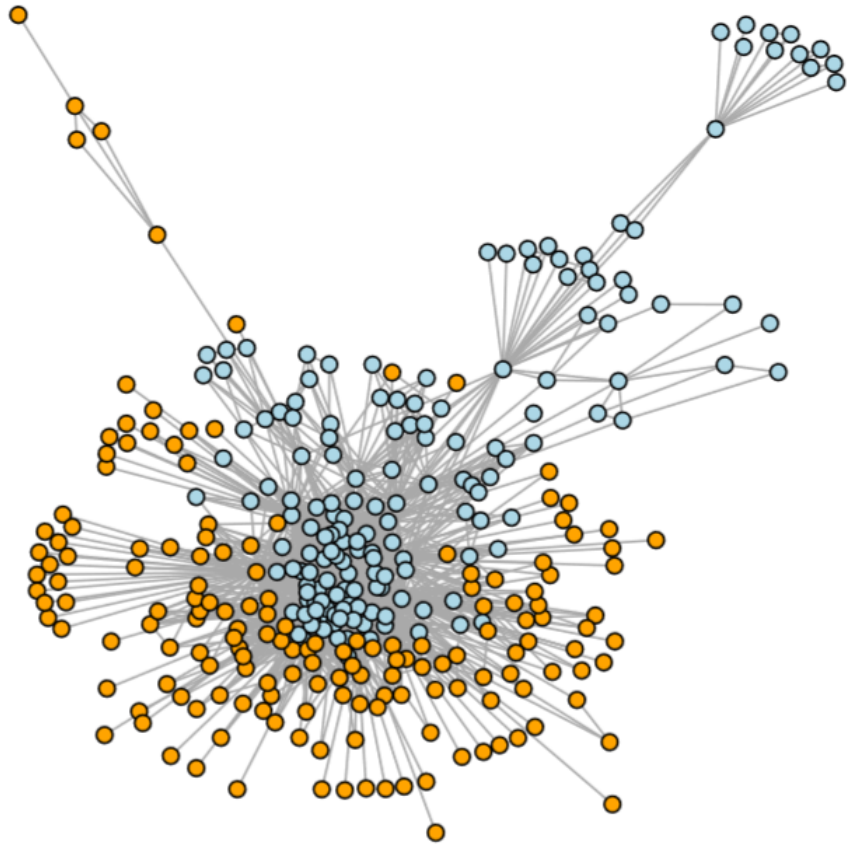

Figura 3.7: Detecção de Comunidades pelo método do Corte com k elementos na comunidade.

Nesse experimento é muito difícil dizer o que seria uma boa partição/comunidade, devido ao tamanho do grafo e a falta de informações *a priori*. Contudo, o método do Corte Mínimo Livre aparenta, pela proximidade dos vértices talvez, ter detectado uma comunidade consistente.

Capítulo 4

Conclusões

Existem diversas áreas em que o conceito de rede é importante, como por exemplo na biologia, em complexos de proteínas, nas redes sociais, redes de transportes, dentre muitas outras. A maioria das redes podem ser associadas a um grafo, estrutura a qual é representada por vértices, pontos que representam elementos da rede e arestas, que representam a relação entre dois elementos, ou até mesmo de um elemento com ele mesmo.

Uma propriedade que geralmente está associada às redes é a existência de comunidades, ou seja, a partição dos vértices da rede em grupos onde a conectividade é mais frequente. A identificação dessas comunidades é interessante pelo fato de que os elementos dessas comunidade possuem, muitas vezes, características semelhantes. Portanto, um problema muito comum é a detecção dessas comunidades. Existem diversos métodos de se resolver esse problemas, como o método do corte mínimo e a maximização da modularidade que será apresentado neste trabalho.

A Modularidade, definida por Newman (2006) [5] surgiu como uma alternativa para a detecção de comunidades, utilizando a análise espectral dos grafos e a maximização da modularidade. A modularidade é um conceito que tem motivado novas abordagens a problemas em aberto da teoria de grafos, como a determinação de *cliques* máximos. Entretanto, na detecção de comunidades para grafos de ordem muito grande, a falta de esparsidade da matriz de modularidade eleva o custo computacional, o que eventualmente pode impedir o uso desse método.

Contudo, nos dois experimentos realizados o método do corte mínimo livre, sem restrição a ordem das comunidades, mostrou-se visivelmente mais adequado.

Esse trabalho trouxe uma abordagem inicial e espera-se que a partir dele possasse seguir em diversas direções como:

- detecção de mais do que duas comunidades, que envolve uma otimização inteira adicional na determinação do vetor de escolha.
- considerar conectividade de maior ordem entre os vértices.
- otimizar a implementação para resolver problemas com grafos de grandes ordens.
- dentro da definição de modularidade, propor novas escolhas para a matriz P.

- quantificar a qualidade da detecção das comunidades, através dos valores das funções Corte e modularidade, dentre outras possíveis.

Bibliografia

[1] C. J. Alpert and S.-Z. Yao, Spectral partitioning: The more eigenvectors, the better. In B. T. Preas, P. G. Karger, B. S. Nobandegani, and M. Pedram (eds.), Proceedings of the 32nd International Conference on Design Automation, pp. 195–200, Association of Computing Ma- chinery, New York, NY (1995).

[2] LUSSEAU, D., SCHNEIDER, K., Boisseau, O. J., HAASE, P., SLOOTEN, E., DAWSON, S. M., The Bottlenose Dolphin Community of Doubtful Sound Features a Large Proportion of Long-Lasting Associations: Can Geographic Isolation Explain This Unique Trait?, **Behavioral Ecology and Sociobiology**, 54,4, 396–405, 2003.

[3] MESBAHI, M., EGERSTEDT, M., Graph theoretic methods in multiagent networks, Princenton University Press, 2010.

[4] N. Abreu, R. R. Del-Vecchio, V. Trevisan e C. T. M. Vinagre. Teoria Espectral de Grafos - Uma Introdução. In Notas do III^o Colóquio de Matemática da Regi ao Sul, Florianópolis, Santa Catarina, Brasil, 2014.

[5] NEWMAN, M.E.J., Finding community structure in networks using eigenvectors of matrices. **arXiv**, arXiv:physics/0605087v3, 2006.

[6] ROSSI, R. A., AHMED, N. K., The Network Data Repository with Interactive Graph Analytics and Visualization, **Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence**, *http://networkrepository.com*, 2015.

[7] S. White and P. Smyth, A spectral clustering approach to finding communities in graphs. In H. Kargupta, J. Srivastava, C. Kamath, and A. Goodman (eds.), Proceedings of the 5th SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, Philadelphia (2005).

[8] SCHAUB, M.T., DELVENNE, J.-C., ROSVALL, M., LAMBIOTTE, R., The many facets of community detection in complex networks, **Applied Network Science**. 2:4. 2017.

Apêndice

4.1 Códigos em R

4.1.1 Código do Problema dos Golfinhos

```
    library(igraph)
dolphin<-graph(edges=c(11, 1, 15, 1, 16, 1, 41, 1,..., 58, 55),directed = FALSE)
V(dolphin)$size <- 7 #o tamanho default eh 15
V(dolphin)$shape<-"circle"
V(dolphin)$shape[c(2,6,7,8,10,14,18,20,23,26,27,28,32,33,42,49,55,57,58,61)]<-"square"
plot(dolphin)
coords<- layout.fruchterman.reingold(dolphin)
#salva o layout do grafo, como plotado na linha anterior
L<-laplacian_matrix(dolphin)
A<- as_adjacency_matrix(dolphin)


#### Particao usando modularidade
D=degree(dolphin);
n=gorder(dolphin);# numero de vertices
m=gsize(dolphin);# numero de arestas
P=(1/(2*m))*D%*%t(D);
B=A-P;#matriz de modularidade
ev<-eigen(B)
Evalues=ev[[1]];# autovalores da matriz de modularidade em ordem decrescente
Evectors<-ev[[2]]# autovetores da matriz de modularidade
s<-matrix(0,n,1)# vetor de escolha
for(i in 1:n){
  if(Evectors[i,1]>=0) s[i,1]=1 else s[i,1]=-1
  #preenche o vetor de escolha conforme os sinais das
  entradas o autovetor associado ao maior autovalor
}
V(dolphin)$color <- ifelse(s[,1] == 1, "lightblue", "orange")
plot(dolphin,layout=coords)
```

```
#### Particao usando corte minimo (livre)
ev1<-eigen(L)
Evalues1=ev1[[1]];# autovalores da matriz de modularidade em ordem decrescente
Evectors1<-ev1[[2]]# autovetores da matriz de modularidade
s1<-matrix(0,n,1)# vetor de escolha
for(i in 1:n){
  if(Evectors1[i,n-1]>0) s1[i,1]=1 else s1[i,1]=-1
  #preenche o vetor de escolha conforme os sinais das
  entradas o autovetor associado ao maior autovalor
}
V(dolphin)$color <- ifelse(s1[,1] == 1, "lightblue", "orange")
plot(dolphin,layout=coords)


#### Particao usando corte minimo
(comunidade 1 com k vértices e comunidade 2 com n-k vértices) ###
k=32
ev2<-eigen(L)
Evalues2=ev2[[1]];# autovalores da matriz de modularidade em ordem decrescente
Evectors2<-ev2[[2]]# autovetores da matriz de modularidade
s2<-matrix(-1,n,1)# vetor de escolha
ordem<-order(Evectors2[,n-1],decreasing = TRUE)
for(i in 1:k){
    s2[ordem[i],1]=1
}
V(dolphin)$color <- ifelse(s2[,1] == 1, "lightblue", "orange")
plot(dolphin,layout=coords)
```

4.1.2 Código das Linhas Aéreas Norte Americanas

```
   library(igraph)
USAir<-graph(edges=c(2, 1, 4, 1, 8, 1, ... 328, 330, 329),directed = FALSE)
V(USAir)$size <- 8 #o tamanho default eh 15
plot(USAir)
coords<- layout.fruchterman.reingold(USAir)
#salva o layout do grafo, como plotado na linha anterior
L<-laplacian_matrix(USAir)
A<- as_adjacency_matrix(USAir)
```

```
#### Particao usando modularidade
D=degree(USAir);
n=gorder(USAir);# numero de vertices
m=gsize(USAir);# numero de arestas
P=(1/(2*m))*D%*%t(D);
B=A-P;#matriz de modularidade
ev<-eigen(B)
Evalues=ev[[1]];# autovalores da matriz de modularidade em ordem decrescente
Evectors<-ev[[2]]# autovetores da matriz de modularidade
s<-matrix(0,n,1)# vetor de escolha
for(i in 1:n){
  if(Evectors[i,1]>=0) s[i,1]=1 else s[i,1]=-1
  #preenche o vetor de escolha conforme os sinais das
  entradas o autovetor associado ao maior autovalor
}
V(USAir)$color <- ifelse(s[,1] == 1, "lightblue", "orange")
plot(USAir,layout=coords)

#### Particao usando corte minimo (livre)
ev1<-eigen(L)
Evalues1=ev1[[1]];# autovalores da matriz de modularidade em ordem decrescente
Evectors1<-ev1[[2]]# autovetores da matriz de modularidade
s1<-matrix(0,n,1)# vetor de escolha
for(i in 1:n){
  if(Evectors1[i,n-1]>0) s1[i,1]=1 else s1[i,1]=-1
  #preenche o vetor de escolha conforme os sinais das
  entradas o autovetor associado ao maior autovalor
}
V(USAir)$color <- ifelse(s1[,1] == 1, "lightblue", "orange")
plot(USAir,layout=coords)


#### Particao usando corte minimo
(comunidade 1 com k vÃ©rtices e comunidade 2 com n-k vértices) ###
k=166
ev2<-eigen(L)
Evalues2=ev2[[1]];# autovalores da matriz de modularidade em ordem decrescente
Evectors2<-ev2[[2]]# autovetores da matriz de modularidade
s2<-matrix(-1,n,1)# vetor de escolha
ordem<-order(Evectors2[,n-1],decreasing = TRUE)
```

```
for(i in 1:k){
  s2[ordem[i],1]=1
}
V(USAir)$color <- ifelse(s2[,1] == 1, "lightblue", "orange")
plot(USAir,layout=coords)
```