Dissertação de Mestrado

Métodos para construção de Árvores Filogenéticas a partir de Redes Metabólicas

Phelipe Araujo Fabres

Orientador: Prof. Dr. Fábio Henrique Viduani Martinez

Faculdade de Computação
Universidade Federal de Mato Grosso do Sul
Ciência da Computação

Março de 2013

Resumo

Desde a concepção da teoria da evolução, a determinação das relações evolutivas entre os organismos existentes tem sido um dos grandes desafios das Ciências Biológicas. Após o advento do sequenciamento de genomas de organismos, árvores filogenéticas passaram a ser concebidas baseadas na similaridade das sequências de pequenas subunidades ribossômicas ou de genes individuais. Com o avanço das técnicas de sequenciamento, uma grande quantidade de informações está atualmente disponível para consulta e análise e muitos métodos têm sido propostos para construção de árvores filogenéticas a partir de características do genoma completo, tais como a composição de oligonucleotídeos, a ocorrência de fragmentos do genoma e a presença/ausência de características metabólicas. Ao mesmo tempo, muitos trabalhos têm sido direcionados no sentido da análise da similaridade dos processos metabólicos dos organismos, já que tais processos estão fortemente relacionados ao ambiente e à adaptação e manutenção do equilíbrio dos componentes de seu meio. Dessa forma, considerações sobre o metabolismo dos organismos podem revelar informações importantes sobre a interação entre os organismos e o ambiente onde vivem, como por exemplo simbiose ou adaptação a ambientes extremos. Neste trabalho, estudamos dois tipos de métodos para construção de árvores filogenéticas baseadas em redes metabólicas. No primeiro deles, uma via metabólica é fixada, as distâncias entre pares de vias metabólicas dos organismos são computadas e, finalmente, uma árvore filogenética é obtida a partir dessas distâncias. No segundo método, um conjunto de vias metabólicas é fixado e, para cada via, o mesmo processo é realizado, isto é, uma árvore filogenética é construída baseada nas distâncias entre as vias metabólicas nesses organismos. Em seguida, o conjunto de árvores assim obtidas é transformado em uma única árvore filogenética dos organismos de entrada.

Palavras-chave: Redes metabólicas, Árvores filogenéticas, Organismos.

Abstract

Since the conception of the evolutionary theory, the determination of evolutionary relations between existing organisms has been a major challenge of the Biological Sciences. After the advent of sequencing genomes of organisms, phylogenetic trees came to be conceived based on similarity of the sequences of small ribosomal subunits or single genes. With the advances of the sequencing techniques, a large amount of information is currently available for query and analysis, and many methods have been proposed for phylogenetic tree reconstruction from characteristics of the complete genome such as the composition of oligonucleotides, the occurrence of genome fragments and the presence / absence of metabolic characteristics. Meanwhile, many studies have been directed towards the similarity analysis of metabolic processes of the organisms, since such processes are strongly related to the environment and to adaptation and balance maintain of components of their environment. Therefore, considerations on metabolism of organisms can reveal important information about the interaction between organisms and the environment where they live, for example symbiosis and adaptation to extreme environments. This work study two types of methods for reconstructing phylogenetic trees of metabolic networks. In the first, one pathway is fixed, distances between pairs of metabolic pathways of the organisms are computed and finally a phylogenetic tree is obtained from these distances. In the second method, a set of metabolic pathways is fixed and for

each pathway the same process is performed, i.e. a phylogenetic tree is constructed based on distances between the metabolic pathways in these organisms. The whole tree thus obtained is transformed into a single phylogenetic tree of the input organisms.

Keywords: Metabolic networks, Phylogenetic trees, Organisms

Sumário

1 Introdução **1**

2 Fundamentação teórica **3**

2.1 Grafos . . . 3

2.1.1 Algoritmo Kuhn (1955) & Munkres (1957) . . . 5

2.2 Biologia . . . 9

2.2.1 Redes e vias metabólicas . . . 9

2.3 Métodos para construção de árvores filogenéticas . . . 10

2.3.1 Neighbor Joining . . . 10

2.3.2 Métodos baseados em quartetos . . . 15

3 Método de Heymans & Singh (2003) **21**

3.1 Algoritmo Heymans & Singh (2003) . . . 21

3.2 Obtenção do grafo de enzimas . . . 22

3.3 Obtenção da similaridade entre cada par de grafos de enzimas . . . 23

3.4 Construção da árvore filogenética . . . 32

3.5 Experimentos e implementação . . . 32

3.5.1 Árvores filogenéticas obtidas . . . 33

4 Método de Zhang et al. (2006) **39**

4.1 Algoritmo . . . 39

4.2 Obtenção do grafo via . . . 39

4.3 Obtenção do grafo base . . . 40

4.4 Conjunto de organismos O_k que contém uma via metabólica p_k . . . 40

4.5 Árvores filogenéticas T_k e árvore filogenética final . . . 41

4.6 Considerações biológicas e experimentos . . . 41

5 Considerações finais **45**

Referências bibliográficas **47**

Capítulo 1

Introdução

A história evolucionária de um grupo de organismos vivos, espécies ou populações, ou de todos os organismos vivos, é um assunto que tem instigado leigos e cientistas há muito tempo. A evolução das espécies é dada por modificações nas populações no decorrer do tempo, podendo separá-las, reuní-las ou extinguí-las. As relações de parentesco, descendência e ancestralidade entre espécies revelam então informações importantes para diversas áreas das Ciências Biológicas e podem ser visualizadas em uma árvore filogenética. No entanto, para construir o passado evolutivo de um conjunto de organismos, temos disponíveis apenas informações no presente, ou seja, os organismos no presente. Assim, nosso conhecimento de como a evolução ocorre nos permite construir uma árvore filogenética [27, 42, 44].

Antes do advento do sequenciamento de genomas de organismos, uma tal história evolutiva era construída a partir das características físicas dos organismos. Depois dele, métodos para construção de árvores filogenéticas se multiplicaram e são os mais populares atualmente [27, 42, 44]. Em um método como esse, as espécies podem estar representadas por suas sequências de DNA, por suas estruturas de RNA e, mais recentemente, por suas redes metabólicas. Uma rede metabólica é o conjunto dos compostos bioquímicos, reações bioquímicas e enzimas que determinam a manutenção de um organismo vivo. Os compostos bioquímicos são importados, exportados, sintetizados ou degradados em um organismo através de reações bioquímicas que, em geral, são disparadas por aceleradores bioquímicos chamados de enzimas. Mais recentemente, métodos para construção de árvores filogenéticas a partir das redes metabólicas dos organismos têm tomado grande atenção da comunidade científica [22, 31, 30, 34, 35, 49].

Neste trabalho, estudamos dois métodos para construção de árvores filogenéticas a partir de redes metabólicas: o método de Heymans & Singh [34, 35] e o método de Zhang *et al.* [63].

O método de Heymans & Singh [34, 35] recebe um conjunto de redes metabólicas de organismos e constroi uma árvore filogenética desses organismos a partir das distâncias entre as redes metabólicas. Um complexo conjunto de fórmulas é responsável por computar a distância entre duas redes metabólicas de dois organismos. Com a matriz de distâncias assim obtida, um método para construção de árvores filogenéticas baseado em distâncias é usado para produzir a árvore filogenética resultante.

O método Zhang *et al.* [63] recebe um conjunto de organismos e suas vias metabólicas e constroi uma coleção de árvores filogenéticas desses organismos para cada via metabólica. Em seguida, usa o método dos quartetos [9], tomando essa coleção de árvores filogenéticas como entrada, para construir uma árvore filogenética global de todos os organismos de entrada.

No capítulo 2, apresentamos a fundamentação teórica computacional e biológica para este trabalho. No capítulo 3, dissertamos sobre o método de Heymans & Singh [34, 35], incluindo uma implementação dele e alguns experimentos sobre um grupo de organismos. A árvore filogenética assim obtida é comparada com a árvore filogenética baseada nas sequências bem conservadas do 16S rRNA dos mesmos organismos. No capítulo 4 apresentamos um estudo teórico do método de Zhang *et al.* [63]. Por fim, no capítulo 5 apresentamos as considerações finais.

Capítulo 2

Fundamentação teórica

Este capítulo acomoda as definições e propriedades fundamentais da Biologia Molecular, da Bioquímica e da Ciência da Computação usadas neste trabalho, baseadas especialmente nas referências bibliográficas [14, 28, 34, 35, 42, 2, 43].

2.1 Grafos

Um **grafo** é uma estrutura discreta composta por **vértices** e **arestas**. Cada aresta é um par não-ordenado de vértices. Uma aresta $\{u, v\}$ de um grafo será denotada simplesmente por uv. Diremos que a aresta uv **incide** em u e em v; diremos também que u e v são as **pontas** da aresta; diremos, ainda, que os vértices u e v são **vizinhos** ou **adjacentes**. Um grafo com conjunto de vértices V e conjunto de arestas E é denotado por (V, E). Em geral, damos um nome para um grafo, como por exemplo $G = (V, E)$. Dizemos que V_G representa o conjunto de vértices V do grafo G e E_G representa o conjunto de arestas E do grafo G.

A **vizinhança de um vértice** v em um grafo G é o conjunto de todos os vizinhos de v. Este conjunto é denotado por $\mathrm{N}_G(v)$ ou simplesmente por $\mathrm{N}(v)$, quando o grafo em questão estiver subentendido. O **grau** de um vértice v em um grafo G é o número de arestas que incidem em v e é denotado por $\mathrm{d}_G(v)$ ou simplesmente por $\mathrm{d}(v)$, quando o grafo em questão estiver subentendido. É fácil ver que $\mathrm{d}(v) = |\mathrm{N}(v)|$ para todo vértice v.

Um grafo G é dito um **grafo com custos nas arestas** se existe uma função $c\colon E \to \mathbb{Q}$ chamada de **função custo nas arestas**. Denotamos G por $G = (V, E, c)$. Se X é um subconjunto de arestas de G então definimos $c(X) = \sum_{e \in X} c(e)$. O custo do grafo G, denotado por $c(G)$ é definido como $c(G) = c(E_G)$.

Uma bipartição de um conjunto V é um par $\{U, W\}$ de conjuntos tal que $U \cup W = V$ e $U \cap W = \emptyset$. Uma **bipartição de um grafo** G é uma bipartição $\{U, W\}$ de V tal que toda aresta de G tem uma ponta em U e outra em W. Um grafo é **bipartido** se estiver munido de uma bipartição.

Duas arestas são **adjacentes** se possuem uma ponta em comum. Um **emparelhamento** é um conjunto de arestas duas a duas não adjacentes. Um vértice v é dito **saturado** por um emparelhamento se é ponta de uma aresta do emparelhamento. Uma aresta uv é dita **saturada** se ela pertencer a um emparelhamento. Se o grafo em questão tem custos nas arestas, o **custo de um emparelhamento** é a soma dos custos de suas arestas. Um emparelhamento $\mathcal{M}$ é chamado **emparelhamento máximo** se

cada emparelhamento $\mathcal{M}'$ satisfizer $|\mathcal{M}'| \leq |\mathcal{M}|$. Em um grafo bipartido G, um emparelhamento $\mathcal{M}$ é um **emparelhamento perfeito** se todo vértice de G for saturado por $\mathcal{M}$.

Um **dígrafo**, **grafo orientado** ou **dirigido** é aquele composto por um conjunto de **vértices** e um conjunto de **arcos**. Um arco em um dígrafo é um par ordenado de vértices. O primeiro vértice do par é chamado de **ponta inicial** e o segundo vértice é a **ponta final** do arco.

Um dígrafo com conjunto de vértices V e conjunto de arcos A é denotado por (V, A). Em geral, damos um nome para um dígrafo, como por exemplo $G = (V, A)$.

Em um dígrafo G, um arco (u, v) será denotado por $\overrightarrow{uv}$ ou simplesmente por uv. Diremos que o arco uv **vai de** u **a** v. Diremos ainda que o arco uv **sai** de u e **entra** em v. O **grau de saída** de um vértice v é o número de arcos que saem de v e é denotado por $\mathrm{d}^+_G(v)$ ou simplesmente por $\mathrm{d}^+(v)$, quando o grafo em questão estiver subentendido. Definimos $\tilde{\mathrm{d}}^+_G(v) = |V_G| - \mathrm{d}^+_G(v)$. O **grau de entrada** de v é o número de arcos que entram em v e é denotado por $\mathrm{d}^-_G(v)$ ou simplesmente por $\mathrm{d}^-(v)$, quando o grafo em questão estiver subentendido. Definimos $\tilde{\mathrm{d}}^-_G(v) = |V_G| - \mathrm{d}^-_G(v)$. Dizemos que um vértice v é **adjacente a** ou **vizinho de** um vértice u se o par $\{u, v\}$ é um arco, ou seja, se existe um arco que sai de u e entra em v.

Um **subgrafo** de um grafo $G = (V, E)$ é qualquer grafo H tal que $V_H \subseteq V_G$ e $E_H \subseteq E_G$. Um **caminho** em G é qualquer subgrafo C de G cujo conjunto de vértices admite uma permutação $(v_1, v_2, \ldots, v_n)$ tal que

$$\{v_1v_2, v_2v_3, \ldots, v_{n-1}v_n\} = E_C \, .$$

Os **extremos** de um caminho são os vértices que iniciam e terminam o caminho. Seja um emparelhamento $\mathcal{M}$ em um grafo G. Um caminho C em G é **alternante** se suas arestas são, de forma alternada, saturadas e não-saturadas por $\mathcal{M}$. Um caminho alternante C é **aumentador** se ele inicia e termina com um vértice não-saturado e possui pelo menos uma aresta.

Um grafo $G = (V, E)$ é **conexo** se existe um caminho entre qualquer par de vértices $\{u, v\} \in V$. Um **ciclo** em G é um caminho C que inicia com um vértice u e termina com o mesmo vértice u, com $u \in V_G$. O **comprimento** de um caminho C ou de um ciclo é o número de arestas de C. Se um caminho tem comprimento k, ele tem $k + 1$ vértices e se um ciclo tem comprimento k, ele tem k vértices. A **distância** $\mathrm{dist}(u, v)$ denota o comprimento de um menor caminho entre u e v, para $\{u, v\} \in V_G$. O **diâmetro** de um grafo é a maior distância entre quaisquer dois vértices do grafo.

Seja um grafo $G = (V, E)$. Considere $\mathrm{dist}(u, u) = 0$ e $\mathrm{dist}(u, v) = 0$ se não há caminho entre u e v. O **comprimento médio dos caminhos** de G, denotado por $\mu(G)$, é

$$\mu(G) = \frac{1}{|V| \times (|V| - 1)} \times \sum_{u,v \in V} \mathrm{dist}(u, v) \, .$$

Uma **árvore** T é um grafo conexo que não possui ciclos. Uma **folha** em T é qualquer vértice u de T tal que $\mathrm{d}_T(u) = 1$. Em um grafo bipartido G com um emparelhamento $\mathcal{M}$, uma árvore T é **alternante** se a partir de um vértice r, com $r \in V_T$, todos os caminhos de r até as folhas dessa árvore são caminhos alternantes com relação a $\mathcal{M}$.

2.1.1 Algoritmo Kuhn (1955) & Munkres (1957)

O método de Heymans & Singh, explicado no capítulo 3, contém um passo fundamental, em que é necessário obter um emparelhamento de custo máximo em um grafo bipartido. As vias metabólicas da entrada do problema são representadas por seus respectivos dígrafos de enzimas. O algoritmo toma então um par de dígrafos de enzimas da entrada e computa a similaridade entre os dois pela análise par a par dos vértices desses dígrafos. Se os dois vértices são "semelhantes", isto é, se possuem conjuntos de vizinhos de entrada e de saída semelhantes em um sentido que ficará claro no capítulo 3, uma alta pontuação é atribuída a essa dupla de vértices gerando naturalmente um grafo bipartido com custos nas arestas. O método de Heymans & Singh usa esse grafo bipartido para calcular a similaridade entre os dois dígrafos da entrada.

Nesta seção descrevemos o algoritmo de *Kuhn e Munkres* [41, 48], que recebe como entrada um grafo bipartido completo com custo nas arestas e devolve um emparelhamento de custo máximo neste grafo.

Conceitos e descrição do algoritmo

Seja um grafo bipartido completo $G = (V_G, E_G, c)$, com $V_G = \{U, W\}$ e c uma função custo nas arestas de G. Uma **rotulação** dos vértices de G é uma função $\ell : V_G \to \mathbb{Q}$. Chamamos essa rotulação de **viável** se

$$\ell(u) + \ell(w) \geq c(uw)\ ,\ \text{para todo } u \in U \text{ e } w \in W\ .$$

Dizemos que G_ℓ é um subgrafo gerador com rótulos nos vértices iguais aos pesos das arestas de G se, para a rotulação ℓ, G_ℓ é tal que $V_{G_\ell} = V_G$ e

$$E_{G_\ell} = \{uw : \ell(u) + \ell(w) = c(uw)\}\ ,\ \text{para todo } u \in U \text{ e } w \in W\ .$$

O teorema 1 é utilizado pelo algoritmo e sua prova pode ser encontrada em [14].

Teorema 1 (Kuhn & Munkres [41, 48])
Seja ℓ uma rotulação viável em G. Se o subgrafo gerador G_ℓ contém um emparelhamento perfeito $\mathcal{M}$ então $\mathcal{M}$ é um emparelhamento de custo máximo em G.

A ideia do algoritmo é encontrar um emparelhamento perfeito de custo máximo. Para isso, ele utiliza uma rotulação viável inicial nos vértices de G e determina o subgrafo gerador G_ℓ e um emparelhamento $\mathcal{M}$ de cardinalidade máxima em G_ℓ como será detalhado na seção 2.1.1. Se $\mathcal{M}$ é um emparelhamento perfeito, o teorema 1 garante que $\mathcal{M}$ é um emparelhamento de custo máximo.

Se $\mathcal{M}$ não é um emparelhamento perfeito, iterativamente o algoritmo tenta aumentar o tamanho de $\mathcal{M}$, procurando um caminho aumentador $P = u, \ldots, y$ com relação a $\mathcal{M}$. Para isso, seja $\mathcal{M}'$ o conjunto de arestas de P que pertencem a $\mathcal{M}$, e $\mathcal{M}'' = E_P - \mathcal{M}'$. Temos então o conjunto $\mathcal{M}_1 = (\mathcal{M} - \mathcal{M}') \cup \mathcal{M}''$ e se observa que $\mathcal{M}_1$ é um emparelhamento para G de cardinalidade $|\mathcal{M}| + 1$.

Se $\mathcal{M}$ não pode ser aumentado, uma nova rotulação viável ℓ é calculada, G_ℓ é reconstruído e o processo é repetido até que o emparelhamento $\mathcal{M}$ seja perfeito. O processo também para se todos os vértices de G sejam saturados. Veja o algoritmo KUHN-MUNKRES.

ALGORITMO KUHN-MUNKRES

Entrada: recebe um grafo bipartido completo $G = (V, E)$ com $V = \{U, W\}$, uma função custo $c : E \to \mathbb{Q}$
Saída: devolve um emparelhamento de custo máximo $\mathcal{M}$

1: seja uma rotulação viável ℓ em G
2: seja o subgrafo gerador G_ℓ de G com conjunto de arestas E_ℓ
3: aplique o algoritmo para obter um emparelhamento de cardinalidade máxima $\mathcal{M}$ (seção 2.1.1)
4: **enquanto** todos os vértices de U não estão saturados em relação a $\mathcal{M}$ **faça**
5: selecione o primeiro vértice $x \in U$ não-saturado
6: construa uma árvore alternante T com relação a $\mathcal{M}$ tal que $x \in U$ é a raiz de T
7: **enquanto** um caminho aumentador P não for descoberto **E** existirem vértices de U não-saturados **faça**
8: atualize os rótulos da seguinte maneira:

$$m_\ell = \min\{\ell(u) + \ell(w) - c(uw) : u \in U \cap V(T) \text{ e } w \in W - V(T)\}$$

$$\ell'(v) = \begin{cases} \ell(u) - m_\ell\,, & \text{para } u \in U \cap V(T) \\ \ell(w) + m_\ell\,, & \text{para } w \in W \cap V(T) \\ \ell(v)\,. & \text{caso contrário} \end{cases}$$

9: $\ell \leftarrow \ell'$
10: reconstrua G_ℓ
11: construa uma árvore alternante T com relação a $\mathcal{M}$ tal que $x \in U$ é a raiz de T e x não é saturado
12: **se** um caminho aumentador P foi descoberto **então**
13: aumente $\mathcal{M}$ sobre P
14: **devolva** $\mathcal{M}$

A respeito da complexidade, toda vez que um árvore alternante é construída $O(|V^2|)$(linha 6) operações são realizadas. A construção de uma árvore alternante é repetida V vezes no pior caso pois todos os vértices de G poderão originar uma árvore alternante, logo teremos que verificar V vezes se a árvore gerada possui ou não um caminho aumentador. Portanto, a estrutura de repetição **enquanto** da linha 7 tem complexidade no pior caso de $O(|V^3|)$. Como a estrutura de repetição **enquanto** da linha 4 é executada no pior caso para todos os V vértices do grafo, temos uma complexidade total para o algoritmo de $O(|V^4|)$.

Emparelhamento de cardinalidade máxima em grafos bipartidos

Um emparelhamento de cardinalidade máxima em um grafo bipartido $G = (V_G, E_G)$ pode ser obtido pela construção de uma sequência de emparelhamentos $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_k$ em G. $\mathcal{M}_1$ é um emparelhamento inicial em G e $\mathcal{M}_i$ é obtido a partir de $\mathcal{M}_{i-1}$ para todo i, $2 \leq i \leq k$, aumentando $\mathcal{M}_{i-1}$ com base em algum caminho aumentador e $\mathcal{M}_k$ é o maior emparelhamento possível, ou seja, $\mathcal{M}_k$ é um emparelhamento de cardinalidade máxima [14].

Para a obtenção dos caminhos aumentadores, construímos uma árvore alternante T, tal que o primeiro vértice $x \in V_G$ não-saturado com relação a $\mathcal{M}$ seja a raiz de T. Se existir um vértice não-saturado u adjacente a x então temos um caminho aumentador. Assim, aumentamos $\mathcal{M}$ sobre esse caminho para

obtermos um emparelhamento de cardinalidade $|\mathcal{M}|+1$. Senão, cada vértice adjacente a x está saturado. Nesse caso, construímos a árvore alternante T com x como raiz e todos os vértices adjacentes a x, digamos $u_1, u_2, \ldots, u_k$, no nível 1, e conectamos x aos vértices u_i para todo i, $2 \leq i \leq k$. Então para $u_i v_i \in \mathcal{M}$ com $1 \leq i \leq k$, colocamos os vértices $v_1, v_2, \ldots, v_k$ no nível 2 da árvore e conectamos u_i com v_i para todo i, $2 \leq i \leq k$.

Digamos que a árvore alternante tenha sido construída até o nível m, onde m é par, e nenhum caminho aumentador começando em x foi encontrado. Para cada vértice v do nível m da árvore, examinamos cada vértice y adjacente a x. Se y já pertence à árvore, nada fazemos. Se y é um vértice saturado e, digamos, $yz \in \mathcal{M}$, então apenas adicionamos os vértices y e z aos níveis $m+1$ e $m+2$. A figura 2.1(a) ilustra essa situação. No entanto, se y for um vértice não-saturado, então um caminho aumentador $P = x, \ldots, y$ é detectado e paramos de construir a árvore alternante. A figura 2.1(b) ilustra essa situação. Aumentamos $\mathcal{M}$ sobre o caminho aumentador P e produzimos um emparelhamento de cardinalidade $|\mathcal{M}|+1$. Se ainda existirem vértices não-saturados com relação a $\mathcal{M}$, repetimos o processo. Senão $\mathcal{M}$ é um emparelhamento máximo. No algoritmo cada vértice possui um atributo que indica se ele está ou não na árvore alternante. Veja o algoritmo ECM.

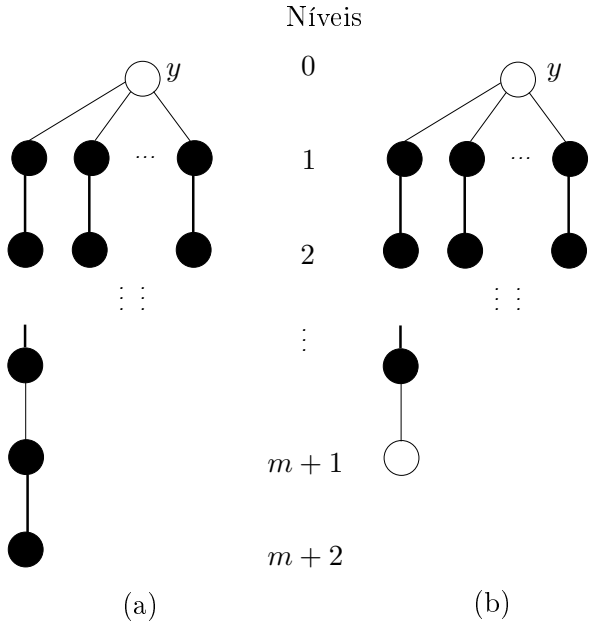

Figura 2.1 Construção da árvore alternante. Em (a), a árvore foi construída até o nível m, m é par e nenhum caminho alternante foi encontrado. Em (b), um caminho alternante foi encontrado e a construção da árvore é interrompida.

Como exemplo de execução, seja G o grafo bipartido mostrado na figura 2.2(a). Um emparelhamento inicial é indicado pelas arestas em destaque. As figuras 2.2(b), 2.3(a) e 2.3(b) mostram como a árvore alternante é construída. Na figura 2.3(b), o caminho alternante $P = x_2, y_6, x_5, y_4, x_1, y_1$ é detectado. Após aumentar o emparelhamento inicial ao longo de P obtêm-se um novo emparelhamento $\mathcal{M}$ que é um emparelhamento de máxima cardinalidade em G.

A respeito da complexidade, a estrutura de repetição **enquanto** da linha 19 do algoritmo ECM depende da fila de vértices Q e é executada, no pior caso, $O(|E|)$ vezes. Dessa forma, como a estrutura de repetição **enquanto** da linha 4 é executada $O(|V|)$ vezes no pior caso, temos que o tempo de execução do algoritmo

ALGORITMO ECM

Entrada: recebe um grafo bipartido completo $G = (V, E)$, com $V = \{U, W\}$ e um emparelhamento inicial $\mathcal{M}_1$

Saída: devolve um emparelhamento de cardinalidade máxima a partir de um emparelhamento inicial $\mathcal{M}_1$

1: $i \leftarrow 1$
2: $\mathcal{M} \leftarrow \mathcal{M}_1$
3: $Q \leftarrow \emptyset$ ▷ fila de vértices
4: **enquanto** existirem vértices $x \in V$ não-saturados com relação a $\mathcal{M}$ **faça**
5: **se** $i > |V|$ **então**
6: **devolva** $\mathcal{M}$
7: **senão**
8: **se** x_i é saturado **então**
9: $i \leftarrow i + 1$
10: **senão**
11: $x \leftarrow x_i$ e Q contém apenas x
12: **para** $j \leftarrow 1, 2, \ldots, |V|$ **e** $j \neq i$ **faça**
13: x_j.in $\leftarrow$ **falso**
14: x_i.in $\leftarrow$ **verdadeiro**
15: **se** $Q = \emptyset$ **então**
16: $i \leftarrow i + 1$
17: **senão**
18: remova um vértice v de Q
19: **enquanto** existir um $y_i \in V_G$ tal que $vy_i \in E_G$ **faça**
20: $j \leftarrow 1$
21: **se** $j \leq k$ **então**
22: $y \leftarrow y_j$
23: **se** y.in $\leftarrow$ **verdadeiro então**
24: $j \leftarrow j + 1$
25: **senão**
26: **se** y é incidente em uma aresta saturada yz **então**
27: y.in $\leftarrow$ **verdadeiro**
28: z.in $\leftarrow$ **verdadeiro**
29: adicione z a Q
30: $j \leftarrow j + 1$
31: **senão**
32: seja um caminho aumentador $P = x, \ldots, y$
33: $\mathcal{M}' \leftarrow (\mathcal{M} \setminus E_P) \cup (E_P \setminus \mathcal{M})$
34: $\mathcal{M} \leftarrow \mathcal{M}'$
35: $i \leftarrow i + 1$

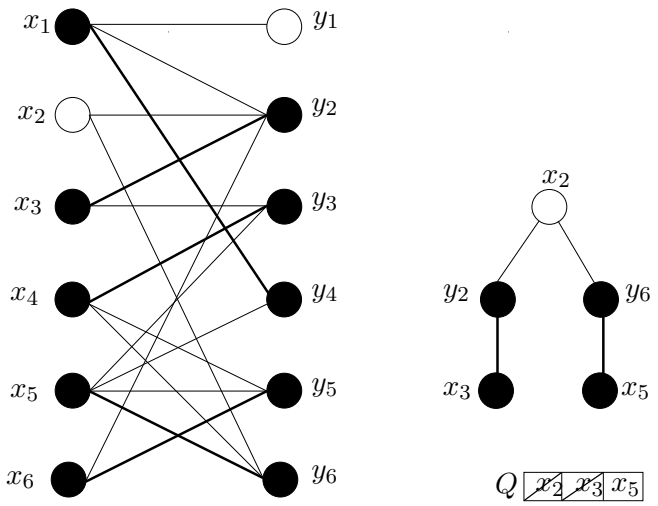

Figura 2.2 Dado um emparelhamento inicial, em (a), o processo de construção da árvore alternante é iniciado encontrando o caminho alternante inicial, em (b).

ECM é $O(|V| \times |E|)$.

2.2 Biologia

2.2.1 Redes e vias metabólicas

Uma **via metabólica** é uma série de reações químicas individuais em um sistema vivo que se combinam para desempenhar uma ou mais funções importantes [35]. Um conjunto de vias metabólicas forma uma rede metabólica.

Uma **rede metabólica** é uma coleção de objetos e das relações entre eles. Os objetos correspondem a compostos químicos, reações químicas e enzimas. **Compostos químicos**, também chamados **metabólitos**, são pequenas moléculas que são importadas/exportadas e/ou sintetizadas/degradadas dentro de um organismo.

Reações químicas produzem um conjunto de um ou mais compostos, chamados **produtos**, a partir de outro conjunto de um ou mais compostos, chamados **substratos**. Dentro de uma célula, a **catalização** de uma reação química é realizada por uma ou mais enzimas que aceleram a velocidade da reação. Uma **enzima** é uma proteína ou um complexo proteico codificado por um ou vários genes.

A figura 2.4 representa a via metabólica do Ciclo de Krebs, que faz parte do conjunto de vias metabólicas que formam a rede do metabolismo do carboidrato.

Número EC é um rótulo numérico aplicado a uma enzima baseado nas reações químicas catalizadas por ela. Dessa forma, números EC não definem ou especificam enzimas, mas sim reações catalisadas por enzimas. Ademais, se enzimas diferentes catalizarem a mesma reação química, então possuem o mesmo número EC.

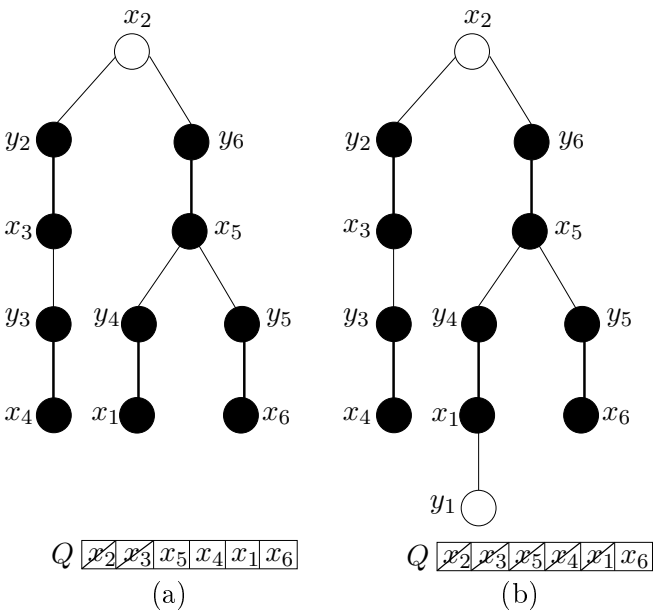

Figura 2.3 Finalização da construção da árvore alternante. Em (b), o caminho será aumentado pela última vez, portanto é o emparelhamento de cardinalidade máxima.

Como exemplo, o número EC *3.4.11.4* representa o rótulo da reação *tripeptide aminopeptidases*. Os dígitos indicados nesse rótulo representam os seguintes grupos de enzimas:

- *3*: hidrolases, isto é, aquelas que usam água para quebrar alguma outra molécula;
- *3.4*: hidrolase que atua sobre ligações peptídicas;
- *3.4.11*: hidrolase que retira o aminoácido amino-terminal de um polipeptídeo;
- *3.4.11.4*: retira o fim do amino-terminal de um tripepitídeo.

2.3 Métodos para construção de árvores filogenéticas

Nos capítulos 3 e 4, diferentes métodos para construir árvores filogenéticas a partir de redes metabólicas serão descritos. Os métodos utilizam um conjunto detalhado de fórmulas matemáticas para encontrar distâncias evolucionárias entre os organismos estudados.

Nesta seção descreveremos um método para construir árvores filogenéticas baseado em matrizes de distância, o *Neighbor Joining*, e um método que reorganiza os organismos e suas árvores em uma grande árvore, o método baseado em quartetos Q^*.

2.3.1 Neighbor Joining

Nos métodos baseados em distâncias ou em matrizes de distâncias, as distâncias evolucionárias são computadas para todos os pares de organismos e uma árvore filogenética é construída considerando as relações entre esses valores [54].

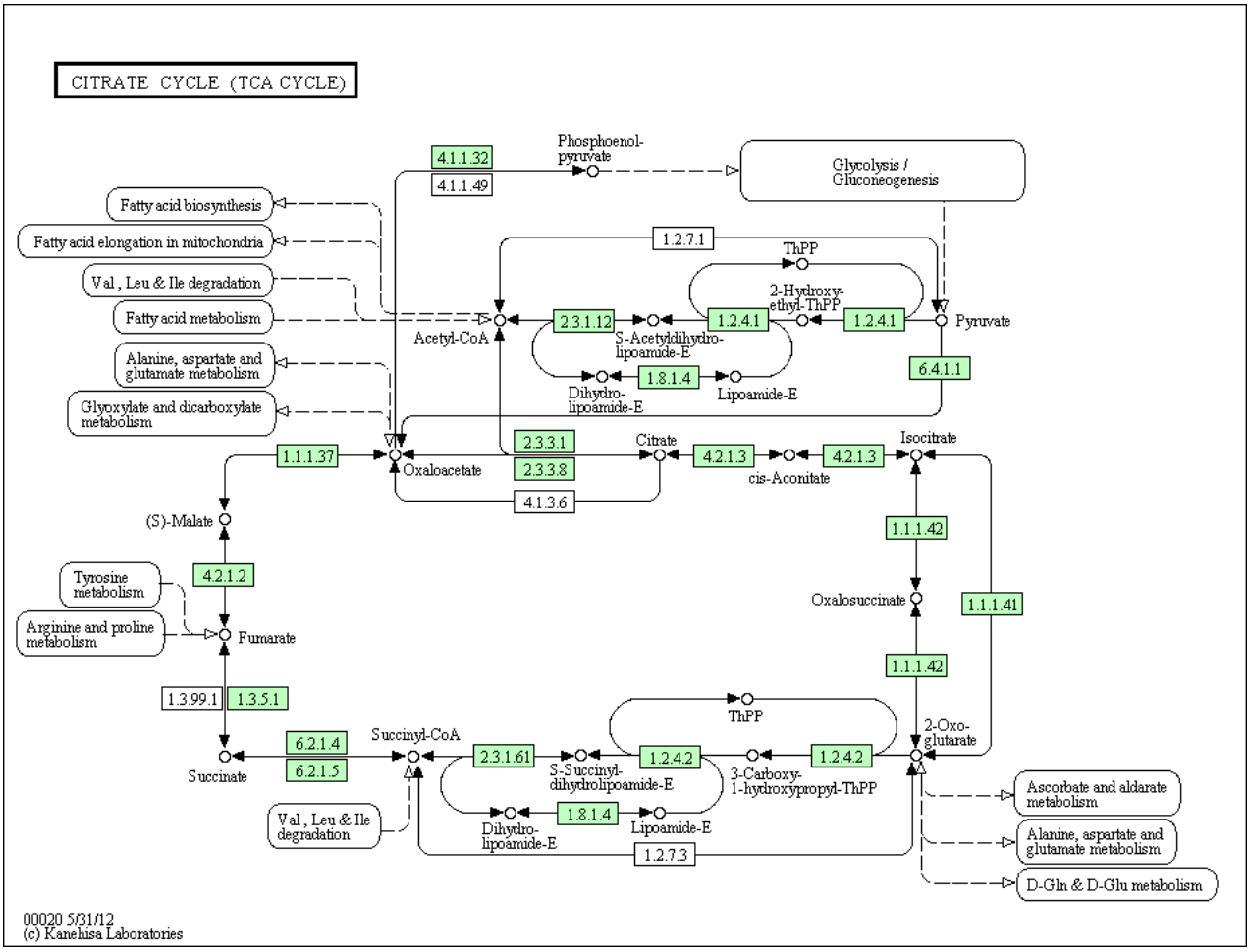

Figura 2.4 Via metabólica do Ciclo de Krebs obtida no repositório do KEEG. As enzimas presentes no Ciclo de Krebs do organismo *Mus Musculus* estão sombreadas.

Neighbor Joining é um método que constrói árvores filogenéticas a partir de matrizes de distâncias seguindo o princípio da evolução mínima (*minimum evolution*) [50, 54]. Este método não examina todas as possibilidades de construção de uma árvore, mas a cada vez que dois organismos são agrupados, o princípio da evolução mínima é usado. O princípio calcula a soma dos comprimentos de todas as arestas de um conjunto de árvores filogenéticas e aquela que possuir a menor soma é considerada a melhor árvore. A cada etapa do processo, o método calcula a árvore cujos comprimentos de suas arestas é mínimo. Nem sempre isso é possível pela dificuldade de se conhecer quais organismos são próximos o suficiente, em termos de evolução, para serem agrupados [54]. Um vértice com grau maior que 1 em uma árvore filogenética é um vértice interno também chamado de **ancestral comum** e a cada folha de uma árvore filogenética é atribuído um organismo da entrada.

Iniciamos a construção de uma árvore filogenética construindo uma árvore-estrela a partir de uma matriz de distâncias de entrada. No passo seguinte, dois organismos serão escolhidos para serem agrupados e a partir desse momento serão representados por seu ancestral comum e não mais individualmente na árvore. Tentamos escolher os organismos cuja a nova árvore formada minimize o valor da soma dos

comprimentos das arestas da árvore. Assim, o processo de agrupamento de organismos é realizado até que restem apenas dois. O método pode ser representado pelos seguintes passos de um algoritmo:

NEIGHBOR JOINING

Entrada: uma matriz de distâncias D de tamanho $m \times m$
Saída: uma árvore filogenética T

1: a partir de D, construa uma árvore-estrela com um vértice interno X e calcule a soma S^* dos comprimentos de suas arestas
2: enquanto existir mais de 2 organismos não agrupados faça
   (a) escolha dois organismos i e j para serem agrupados tal que i e j serão representados agora por um único vértice interno Y, e a nova árvore formada por eles possua o menor valor da soma S_{ij} dos comprimentos de suas arestas
   (b) calcule o comprimento da nova aresta, resultante do agrupamento, que liga o novo vértice Y com o vértice X
   (c) calcule o comprimento das arestas que ligam i e j ao novo vértice Y
   (d) calcule a nova matriz de distâncias agora com tamanho $m - 1 \times m - 1$

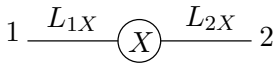

Figura 2.5 Árvore-estrela com 2 organismos.

Como exemplo, seja D uma matriz de distâncias de tamanho $m \times m$ vamos construir uma árvore filogenética T a partir de D usando o *Neighbor Joining*. Inicialmente construiremos uma árvore-estrela, com apenas um único ancestral em comum, o vértice X. Vamos calcular a soma S dos comprimentos das arestas da árvore filogenética, que para a árvore-estrela representaremos por S^*.

A única informação disponível é a matriz de distâncias, ou seja, a distância D_{ij} entre quaisquer pares de organismos i, j. Para uma árvore-estrela com dois organismos, como a da figura 2.5, S^* é dado por

$$S^* = L_{1X} + L_{2X} \, . \tag{2.1}$$

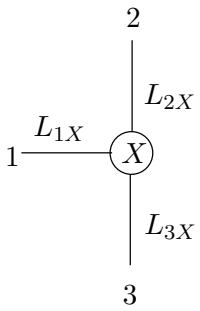

Figura 2.6 Árvore-estrela com 3 organismos.

Embora não conhecemos L_{1X} e L_{2X} é possível deduzir diretamente que $L_{1X} + L_{2X} = D_{12}$, levando a $S^* = D_{12}$. Similarmente, para uma árvore-estrela com três organismos como a da figura 2.6, S^* é dado por $S^* = L_{1X} + L_{2X} + L_{3X}$ e mesmo sem conhecermos L_{1X}, L_{2X} e L_{3X} é possível deduzir que

$$\begin{aligned} S^* &= L_{1X} + L_{2X} + L_{3X} \,, \\ &= \frac{1}{2}\Big\{(L_{1X} + L_{2X}) + (L_{1X} + L_{3X}) + (L_{2X} + L_{3X})\Big\} \,, \\ &= \frac{D_{12} + D_{13} + D_{23}}{2} \,. \end{aligned} \tag{2.2}$$

Seguindo o mesmo raciocínio, para uma árvore-estrela com um número arbitrário $m \geq 2$ de organismos, a soma dos comprimentos de suas arestas é

$$S^* = \sum_{i=1}^{m} L_{iX} = \Big(\frac{1}{m-1}\Big) \sum_{i<j}^{m} D_{ij} \,. \tag{2.3}$$

A partir desse cálculo, precisamos agora escolher dois organismos vizinhos i e j que forneçam uma nova árvore com valor de S menor que o obtido anteriormente. A figura 2.7 exemplifica essa escolha. Um novo vértice interno Y é criado e ligado a X, agora i e j são agrupados como se fossem um único vértice, representado por Y. Seja S_{ij} a soma do comprimento de todas as arestas formadas pela nova árvore com os vizinhos i e j. Esse valor é calculado pela seguinte soma: comprimento das arestas dos dois organismos agrupados mais comprimento da nova aresta surgida entre Y e X mais o comprimento das arestas entre os outros $m-2$ organismos da árvore-estrela e Y.

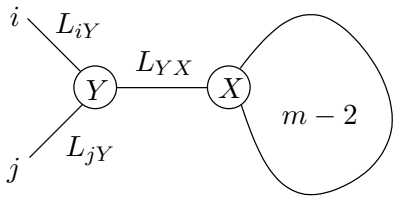

Figura 2.7 Agrupamento de dois organismos i e j. Ao serem agrupados, serão representados por seu ancestral comum, o vértice Y.

O comprimento das arestas dos organismos agrupados é $L_{iY} + L_{jY} = D_{ij}$, como visto anteriormente. Para os p organismos restantes, mesmo sem conhecermos o valor de L_{Xp}, podemos deduzir pelo raciocínio usado anteriormente que o valor de L_{Xp} é

$$\sum_{p=1}^{m} L_{Xp} = \frac{1}{m-3} \sum_{l<k} D_{lk} \,, \tag{2.4}$$

onde p, l, $k \in \{1 \ldots m\} \setminus \{i, j\}$.

Por fim, temos o comprimento L_{YX} da nova aresta surgida entre Y e X. Para chegarmos a esse valor precisamos somar a distância entre os organismos i e j e todos os demais $m-2$ organismos, subtrair dessa soma o comprimento de todas as arestas diferentes de YX e dividir o resultado desta subtração pelo número de vezes que L_{YX} foi somado. Esse valor L_{YX} é dado por:

$$L_{YX} = \frac{1}{2(m-2)} \left(\sum_{k=1}^{m} (D_{ik} + D_{jk}) - (m-2) D_{ij} - 2 \sum_{1 \leq l < k}^{m} D_{lk} \right) , \tag{2.5}$$

onde $l, k \in \{1 \dots m\} \setminus \{i, j\}$.

O valor de S_{ij} é dado por:

$$S_{ij} = \frac{1}{2(m-2)} \sum_{k=1}^{m} (D_{ik} + D_{jk}) + \frac{1}{2} D_{ij} + \frac{1}{m-2} \sum_{1 \leq l < k} D_{lk} \,, \tag{2.6}$$

onde $l, k \in \{1 \dots m\} \setminus \{i, j\}$.

Com S_{ij} podemos escolher os organismos que, para o método, fornecem a melhor árvore naquele momento. O conhecimento das distâncias L_{iY} e L_{jY} não foi necessário para se obter S_{ij}, no entanto o método deve fornecer também o comprimento dessas arestas. Para tanto, seja a figura 2.8 uma representação da figura 2.7 com o vértice Z representando todas as demais $m-2$ folhas da árvore.

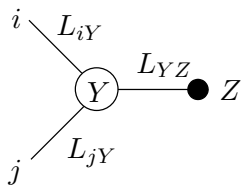

Figura 2.8 Após o agrupamento de dois organismos i e j. Agora precisamos descobrir os comprimentos das arestas entre Y e i e j. O vértice Z representa os outros $m-2$ organismos restantes na árvore.

Vamos então determinar os valores de L_{iY} e L_{jY}. Sabemos que $D_{ij} = L_{iY} + L_{jY}$ e vamos obter as distâncias entre i e j e todos os outros organismos. Temos que $D_{iZ} = L_{iY} + L_{YZ}$ e $D_{jZ} = L_{jY} + L_{YZ}$. Como Z é o vértice auxiliar que representa todas as $m-2$ folhas da árvore, essas distâncias são dadas pela média das distâncias dos organismos i e j a todas as demais folhas da árvore. Assim temos:

$$D_{iZ} = L_{iY} + L_{YZ} = \frac{1}{2(m-2)} \sum_{k=1}^{m} D_{ik} \,, \tag{2.7}$$

$$D_{jZ} = L_{jY} + L_{YZ} = \frac{1}{2(m-2)} \sum_{k=1}^{m} D_{jk} \,, \tag{2.8}$$

onde $k \in \{1 \dots m\} \setminus \{i, j\}$.

Conhecendo essas distâncias, podemos obter os valores de L_{iY} e L_{jY} pelas seguintes equações:

$$L_{iY} = \frac{1}{2(m-2)} \Big((m-2) D_{ij} + \sum_{k=1}^{m} D_{ik} - \sum_{k=1}^{m} D_{jk}) \Big) \,, \tag{2.9}$$

$$L_{jY} = \frac{1}{2(m-2)} \Big((m-2) D_{ij} - \sum_{k=1}^{m} D_{ik} + \sum_{i=1}^{m} D_{jk}) \Big) \,. \tag{2.10}$$

Os organismos i e j da árvore-estrela inicial agora serão representados por seu ancestral comum Y nessa nova árvore. Para que o processo continue, precisamos calcular uma nova matriz de distâncias D' com dimensão $(m-1 \times m-1)$ com Y substituindo i e j. Essas novas distâncias D'_{Yk} entre Y e todos os outros k organismos é dada por

Matriz de Distâncias						
Organismos	1	2	3	4	5	6
1	0	9	12	15	20	16
2	9	0	7	10	15	11
3	12	7	0	5	10	6
4	15	10	5	0	11	7
5	20	15	10	11	0	8
6	16	11	6	7	8	0

Tabela 2.1 *Matriz de distâncias de entrada para a execução do método ilustrado na figura 2.9.*

$$D'_{Yk} = (D_{ik} + D_{jk} - D_{ij})/2\,, \tag{2.11}$$

onde $k \in \{1 \ldots m\} \setminus \{i, j\}$.

Cada vez que a matriz D é recalculada, um novo valor de S é obtido pela fórmula (2.3) para a atual árvore filogenética. Esperamos que esse valor seja menor ou pelo menos igual ao anterior. Por fim, uma árvore filogenética final é obtida quando restarem menos de dois organismos para serem agrupados.

Dada a matriz da tabela 2.1 como entrada, a figura 2.9 ilustra o processo de execução do *Neighbor Joining*, sendo que na figura 2.9(f) temos a árvore resultante.

2.3.2 Métodos baseados em quartetos

Dentre os métodos de construção de árvores filogenéticas existentes, os métodos baseados em quartetos têm recebido atenção especial [15, 37, 33] devido às suas propriedades combinatoriais evidentes, à forma de visualizar a construção de árvores filogenéticas sob a luz do paradigma da divisão e conquista e à facilidade em superar os problemas da disparidade dos dados de entrada [15].

Um **4-subconjunto** de um conjunto de sequências X é um subconjunto de 4 sequências de X. Uma **árvore de um 4-subconjunto**, ou simplesmente um **quarteto**, é uma árvore filogenética totalmente resolvida associada a um 4-subconjunto, isto é, uma árvore binária e sem raiz, com as 4 sequências de um 4-subconjunto de X atribuídas biunivocamente aos seus vértices folhas, com vértices internos com grau 3. Pela definição, um 4-subconjunto $\{a, b, c, d\} \subset X$ tem 3 possíveis quartetos, denotados por $ab|cd, ac|bd$ e $ad|bc$, como mostra a figura 2.10.

Um quarteto $ab|cd$ é **induzido** em uma árvore filogenética T se e somente se $(a \overset{T}{\sim} b) \cap (c \overset{T}{\sim} d) = \emptyset$, onde $(a \overset{T}{\sim} b)$ é o caminho entre as folhas $a, b \in V_T$ na árvore T. Um exemplo de um quarteto induzido em uma árvore filogenética T para um conjunto de sequências X é apresentado na figura 2.11.

O resultado a seguir, devido a [11], motiva o estudo dos métodos baseados em quartetos e ilustra a importância do conjunto de quartetos Q na construção da árvore filogenética T associada. Vale observar ainda que deste resultado decorre imediatamente o fato que uma árvore filogenética T é unicamente determinada por seu conjunto de quartetos induzidos Q.

Teorema 2 *Sejam T^1 e T^2 duas árvore filogenéticas com as folhas rotuladas pelo mesmo conjunto de sequências X. Sejam Q_{T^1} e Q_{T^2} os conjuntos dos quartetos induzidos por T^1 e T^2, respectivamente. Então, $Q_{T^1} = Q_{T^2}$ se e somente se $T^1 = T^2$.*

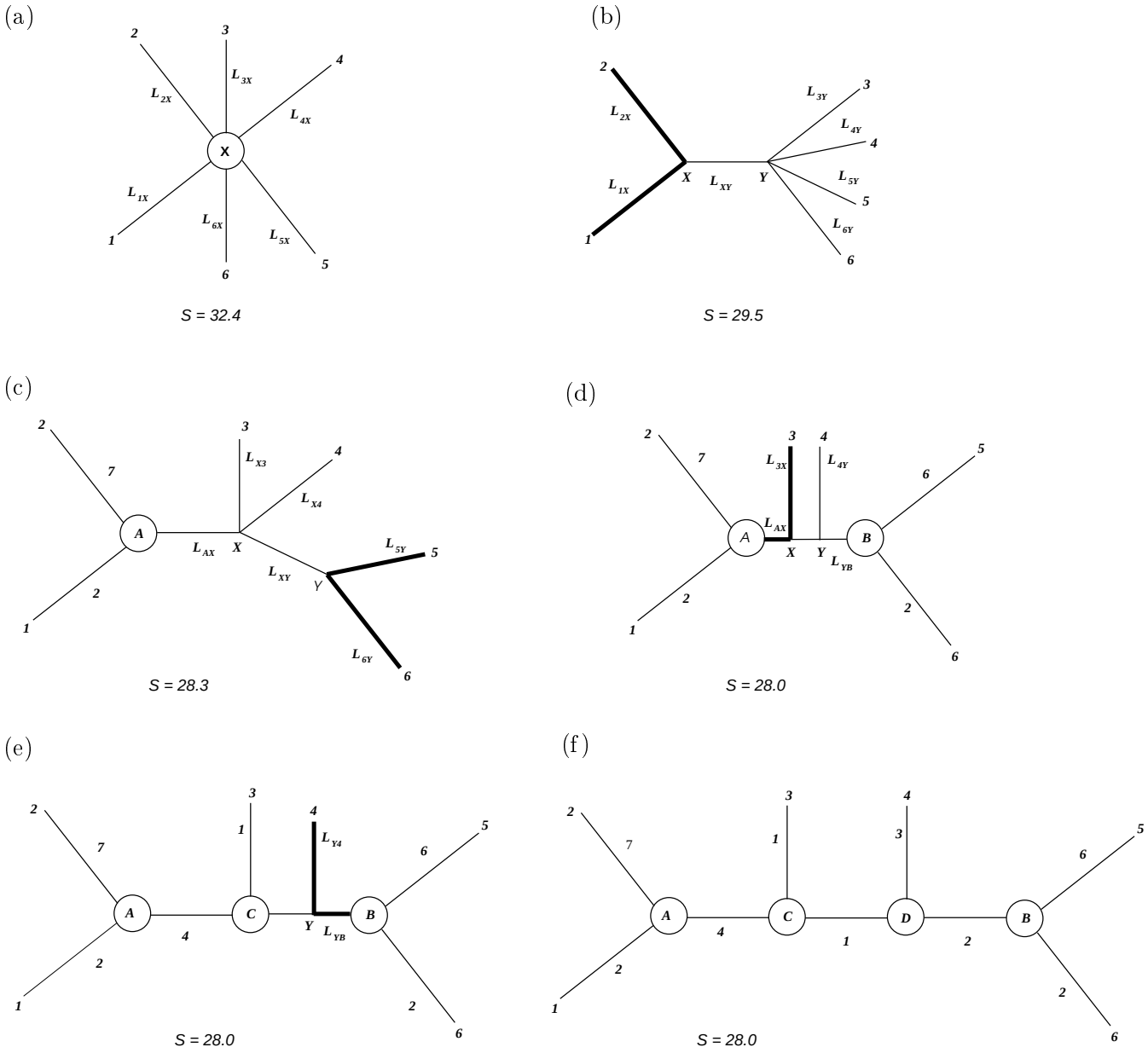

Figura 2.9 Construção da árvore filogenética pelo *Neighbor Joining* tendo a matriz da tabela 2.1 como entrada. As arestas destacadas em cada etapa são as escolhidas para serem unidas, em vários casos existe a união entre um organismo e um vértice interno, que não é um organismo, mas que representa aqueles que já foram unidos. Note que a cada reconstrução da árvore, obtem-se um S menor ou igual ao anterior. Em (f), temos a árvore filogenética resultante.

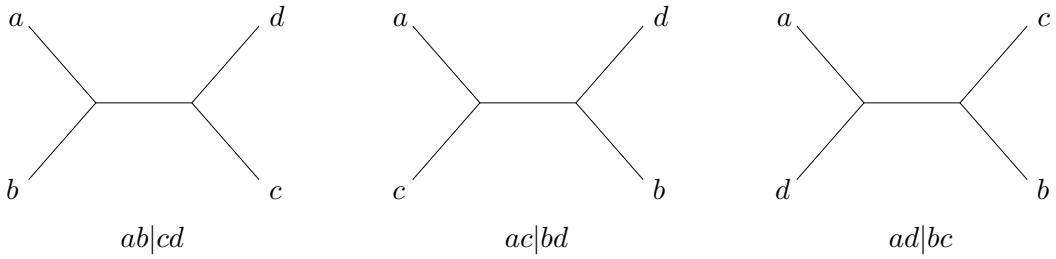

Figura 2.10 Os três possíveis quartetos para o 4-subconjunto $\{a, b, c, d\} \subset X$.

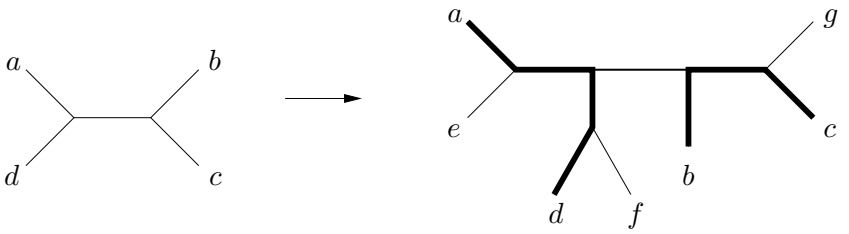

Figura 2.11 Um quarteto induzido em uma árvore.

□

Um método baseado em quartetos tem como objetivo construir uma árvore filogenética T associada a um dado conjunto de quartetos estimados Q. Se o conjunto de quartetos estimados Q é bem definido, então, pelo teorema 2, podemos computar eficientemente a árvore filogenética T a partir de Q [9]. No entanto, o conjunto de quartetos estimados Q nem sempre é bem definido e usualmente possui imperfeições. A construção de uma estimativa da árvore filogenética a partir do conjunto de quartetos estimados pode então ser descrita como um problema de otimização e muitos desses problemas são NP-difíceis, como mencionaremos adiante.

Descrição geral de um método baseado em quartetos

Uma forte motivação para a utilização de métodos baseados em quartetos decorre da propriedade de uma árvore filogenética T ser unicamente caracterizada por seu conjunto de quartetos Q [11], como já mencionado anteriormente.

A estimativa do conjunto de quartetos Q para um conjunto de sequências X é potencialmente realizada por qualquer método de construção de árvores filogenéticas baseado em sequências, indicando outra vantagem importante do emprego de tais métodos. Métodos baseados em distância, parcimônia máxima ou mesmo métodos que consomem mais tempo – como verossimilhança máxima – podem ser utilizados na construção de um quarteto para cada 4-subconjunto de X.

Como também já mencionado acima, outra motivação para a utilização de um método baseado em quartetos é sua capacidade de superar o problema da disparidade dos dados [15]. Esse problema decorre da existência de diferentes quantidades de informação sequenciadas para espécies distintas. Geralmente, um método de construção de árvores filogenéticas recebe como entrada um alinhamento de um conjunto de sequências recuperadas de algum repositório. Entretanto, devido à essa disparidade dos comprimentos

das sequências recuperadas, muita informação é perdida no processo de alinhamento e a efetividade do método é questionada. Um método baseado em quartetos supera essa dificuldade permitindo o uso da maior quantidade de informação possível para cada subconjunto de sequências de entrada.

O conjunto de quartetos Q para o conjunto de sequências em X pode conter quartetos estimados incorretos, comprometendo assim a construção adequada de uma árvore filogenética que induz tais quartetos em Q. Essa estimativa inexata é a principal limitação dos métodos baseados em quartetos. Um exemplo dessa situação é apresentado na figura 2.12.

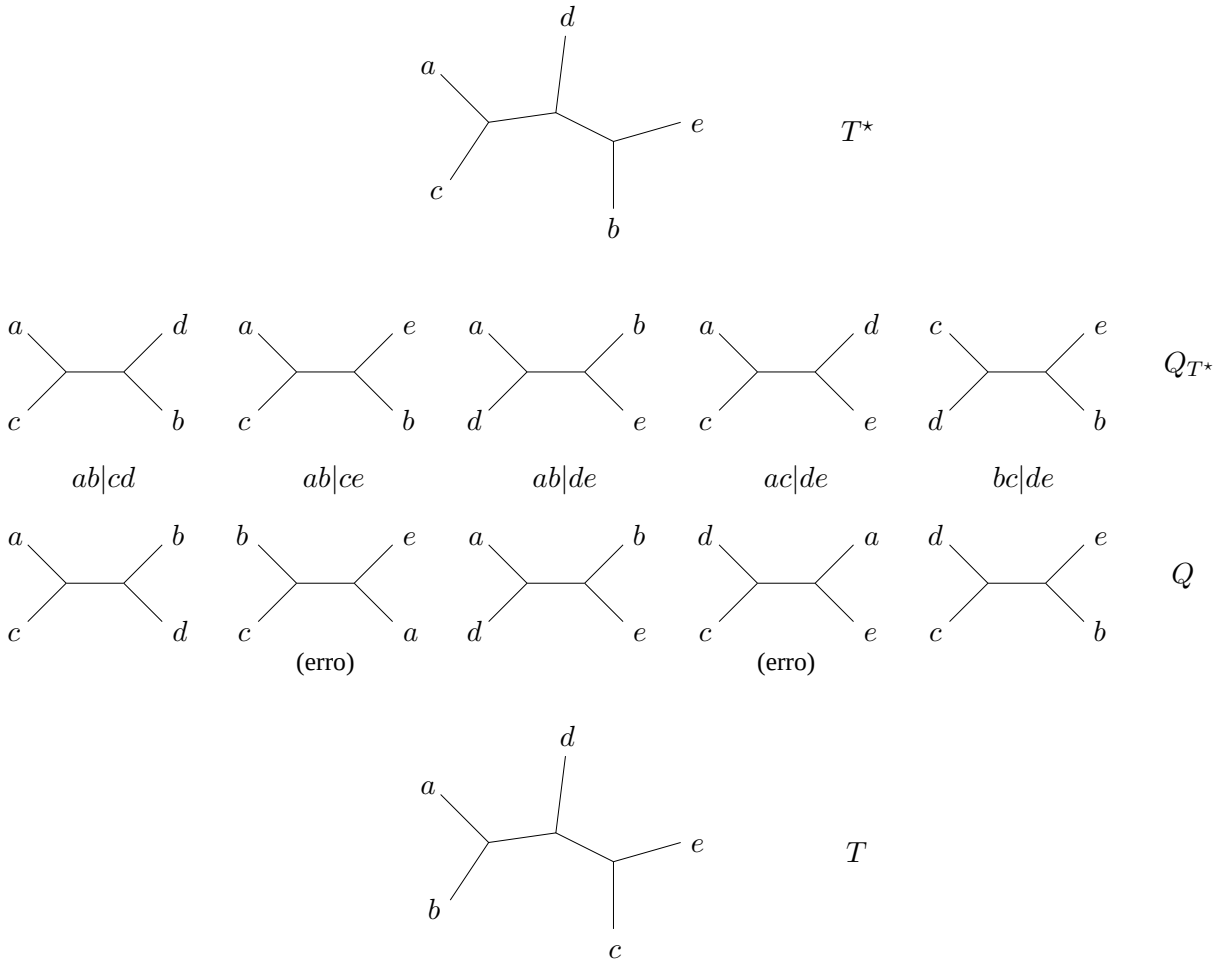

Figura 2.12 Uma árvore $T^\star$, seu conjunto de quartetos induzidos $Q_{T^\star}$, um conjunto de quartetos estimados Q, com os quartetos incorretos destacados, e a árvore filogenética T estimada de Q.

Basicamente, um método baseado em quartetos estima um quarteto para cada 4-subconjunto de X e, em seguida, combina todos os quartetos obtidos na construção de uma árvore filogenética T para X. Portanto, de maneira geral, dois passos são realizados em um método baseado em quartetos:

(i) a inferência de um quarteto para todo 4-subconjunto do conjunto de sequências X;

(ii) a combinação dos quartetos obtidos na construção de uma árvore filogenética T para o conjunto completo de sequências X.

No primeiro passo, um conjunto Q dos $\binom{n}{4}$ quartetos estimados, associados às n sequências de X, é construído. Qualquer método de construção de árvores filogenéticas baseado em sequências – como parcimônia máxima, verossimilhança máxima ou métodos baseados em distâncias – pode ser utilizado para construir o conjunto Q. O passo seguinte consiste da combinação dos quartetos em Q, produzindo uma árvore filogenética estimada T para as n sequências de X.

Rotineiramente, o conjunto estimado de quartetos Q associado ao conjunto de sequências X é incompleto, contém quartetos incorretos ou contém mais de um quarteto para cada 4-subconjunto. Dessa forma, o problema de encontrar uma estimativa de uma árvore T para o conjunto de sequências X baseado no conjunto de quartetos estimados Q torna-se um problema de otimização.

Dado um conjunto de quartetos Q sobre o conjunto de sequências X, um quarteto $ab|cd \in Q$ é um **quarteto incorreto** se $ab|cd \notin Q_{T^\star}$, onde $Q_{T^\star}$ é o conjunto de quartetos induzidos pela árvore filogenética correta $T^\star$. A figura 2.12 mostra uma árvore filogenética correta $T^\star$, seu conjunto de quartetos $Q_{T^\star}$ induzidos por $T^\star$, um conjunto de quartetos estimados Q com seus quartetos incorretos e a árvore filogenética estimada T.

Decorrentes de quartetos incorretos no conjunto de quartetos estimados Q, os conflitos entre quartetos na fase de recombinação podem resultar em imprecisões na determinação de uma árvore filogenética estimada T completamente resolvida e, assim, tal estimativa pode conter vértices internos com grau maior que 3. Neste caso, existem técnicas que determinam um conjunto de arestas completamente suportadas, com algum grau de flexibilidade, que farão parte da árvore filogenética estimada.

Dada uma árvore filogenética T com as folhas rotuladas bijetivamente pelo conjunto de sequências de entrada X, cada aresta $e \in E_T$ de T induz uma **bipartição** $\langle A_e, B_e \rangle$ de X tal que $T - e$ consiste de duas árvores cujos conjuntos de folhas são A_e e B_e. Se $e \in E_T$ é uma aresta de T e $\langle A_e, B_e \rangle$ é sua bipartição correspondente, então a aresta e é chamada uma **aresta completamente suportada** se para todo par de folhas $a, a' \in A_e$ e todo par de folhas $b, b' \in B_e$, o quarteto correspondente ao 4-subconjunto $\{a, a', b, b'\}$ é $aa'|bb'$.

O problema de construir uma árvore filogenética para o conjunto das sequências em X a partir de árvore menores, obtidas no passo da recombinação dos quartetos inferidos, é talvez a principal dificuldade de um método baseado em quartetos. O problema geral de determinar a árvore filogenética T que discorda do menor número de quartetos estimados em Q é NP-difícil [57]. Neste caso, no conjunto Q pode não haver um quarteto para necessariamente todo 4-subconjunto de X.

Inconsistência Mínima de Quartetos Esparsos – IMQ-Esparsos

Instância: um conjunto X de n sequências e um conjunto Q de quartetos associado a X tal que Q pode não conter um quarteto para todo 4-subconjunto de X.

Objetivo: construir uma árvore filogenética T para X tal que $|Q_T \setminus Q|$ é minimizado.

O problema IMQ-Esparsos é NP-difícil [57]. A mesma afirmação é válida se pesos são atribuídos aos quartetos de Q. Se o número de sequências n em X é limitado, o algoritmo exato de [6, 7] tem complexidade de tempo $O(3^n n^4)$ sobre as n sequências e os $O(n^4)$ quartetos.

O problema geral pode ser estudado com a restrição de haver um quarteto em Q para cada 4-subconjunto de X:

Inconsistência Mínima de Quartetos – IMQ

Instância: um conjunto X de n sequências e um conjunto Q de quartetos associado a X tal que existe exatamente um quarteto para todo 4-subconjunto de X.

Objetivo: construir uma árvore filogenética T para X tal que $|Q_T \setminus Q|$ é minimizado.

O problema IMQ também é NP-difícil [57]. Diversas heurísticas para solucionar o problema IMQ vêm sendo propostas por cientistas da computação, matemáticos e biólogos. As primeiras heurísticas [55, 29, 19, 4] utilizam um esquema de pontuação por similaridade de quartetos, ou por vizinhança de quartetos, e constroem uma árvore filogenética T para X com base nessa pontuação.

A heurística de montagem de quartetos (*quartet puzzling*), proposta por Strimmer e von Haeseler em [58], ordena o conjunto de sequências de entrada arbitrariamente, constrói uma árvore inicial para as primeiras quatro sequências dessa ordenação e, em seguida, insere uma sequência por vez na árvore de acordo com uma pontuação sobre seus quartetos. Heurísticas similares, com critérios de pontuação distintos, foram propostas por Willson [61] e Csürös e Kao [20, 21].

A heurística de Ben-Dor *et al.* [6, 7] utiliza programação semidefinida para dispor as n folhas da árvore no $\Re^n$ e estima uma árvore filogenética através de uma busca por vizinhos mais próximos.

O método do quarteto conciso (*short quartet method*), devido a Erdös *et al.* [24, 25, 26], é uma heurística desenvolvida com o objetivo específico de estimar o conjunto dos quartetos Q, realizando uma seleção gulosa dos quartetos inferidos. Em seguida, uma árvore filogenética T para X que procura minimizar $|Q_T \setminus Q|$ é construída.

Para qualquer heurística de construção de árvores filogenéticas baseada em quartetos, como aquelas acima relacionadas, é um problema em aberto provar alguma garantia de desempenho de seu algoritmo correspondente. De fato, o desempenho de algumas heurísticas baseadas em quartetos tem sido recentemente questionado em [56].

Um algoritmo de aproximação com fator de aproximação n^2 foi proposto por Berry *et al.* [10, 8].

Algoritmos exatos de tempo polinomial têm sido propostos para solucionar versões mais restritas do problema IMQ. No trabalho de Buneman [11] há uma caracterização das árvores que se constituem de arestas completamente suportadas. Berry e Gascuel em [9] propuseram algoritmos para recuperar tais árvores: o algoritmo $Q^\star$ com complexidade de tempo $O(n^5)$ e uma versão melhorada com complexidade de tempo $O(n^4)$. Como mencionado em [9], se ao contrário do conjunto de quartetos estimados considerarmos a matriz de distâncias entre as sequências em X como entrada para o problema IMQ, reduzir a complexidade de tempo do algoritmo $Q^\star$ para $O(n^2)$ é uma problema em aberto. Observe que todos os quartetos podem ser inferidos através dessa matriz de distâncias. O método $Q^\star$ foi utilizado por [39] para construção de árvores filogenéticas a partir de um conjunto de quartetos obtido pelo método do quarteto ordinal.

O método de Zhang *et al.* [63], que descrevemos no capítulo 4, usa o algoritmo Q^*, devido a Berry e Gascuel [9], para construir árvores filogenéticas.

Capítulo 3

Método de Heymans & Singh (2003)

A coleção de reações químicas e enzimas que um organismo usa para ativar uma determinada função metabólica determina a arquitetura e a conformação geométrica de uma via ou rede metabólica. Neste capítulo apresentamos o método de Heymans & Singh (2003) [34, 35] que recebe um conjunto de vias ou de redes metabólicas de organismos e constroi uma árvore filogenética desses organismos a partir das distâncias entre as vias ou redes de entrada.

Na seção 3.1 apresentamos o método na forma de um algoritmo. Em seguida, descrevemos os 3 grandes passos do método: obtenção dos grafos de enzimas (seção 3.2), cálculo da similaridade entre cada par de grafos de enzimas (seção 3.3) e construção da árvore filogenética a partir da distância entre as vias ou redes de entradas (seção 3.4).

3.1 Algoritmo Heymans & Singh (2003)

O método de Heymans & Singh (2003) [34, 35], aqui representado na forma de um algoritmo, recebe um conjunto de vias ou de redes metabólicas de organismos e constroi uma árvore filogenética desses organismos a partir das distâncias entre elas. Um complexo conjunto de fórmulas é responsável por computar a distância entre duas vias ou redes metabólicas de dois organismos. Com a matriz de distâncias assim obtida, um método para construção de árvores filogenéticas baseado em distâncias é usado para produzir a árvore filogenética resultante.

Algoritmo Heymans & Singh (2003)

Entrada: recebe duas ou mais redes metabólicas, ou um conjunto de vias metabólicas
Saída: devolve uma árvore filogenética

1: obtenha o grafo de enzima a partir de cada via ou rede metabólica de entrada
2: para cada par de grafos de enzimas G_1 e G_2 obtidos no passo 1 compute a similaridade entre eles:
   (a) compute a similaridade entre cada par de vértices u, v, onde $u \in V_{G_1}$ e $v \in V_{G_2}$, considerando os vértices $x \in V_{G_1}$ e $y \in V_{G_2}$ e as relações de vizinhança desses vértices com u e v
   (b) a partir da similaridade obtida no passo anterior, construa um grafo bipartido com custo nas arestas e compute o emparelhamento de custo máximo
   (c) recompute a similaridade entre os dois grafos G_1 e G_2 pela soma das similaridades dos vértices

saturados pelo emparelhamento obtido no passo 2(b) obtendo assim uma matriz de distâncias entre G_1 e G_2

3: construa uma árvore filogenética usando o método de distâncias *Neighbor Joining* a partir da matriz de similaridades entre os grafos de entrada obtida no passo 2

3.2 Obtenção do grafo de enzimas

Dada uma rede metabólica R, podemos representá-la como um digrafo chamado **grafo de enzimas** $G = (V, E)$, onde o conjunto de vértices V consiste das enzimas presentes em R e o conjunto de arcos E representam os relacionamentos entre as enzimas. Duas enzimas $e_i, e_j \in V_G$ são tais que $e_i e_j \in E_G$ se, e somente se, existe pelo menos um produto da reação catalizada por e_i que é substrato da reação catalizada por e_j. Além disso, os vértices de G, que representam enzimas que catalizam reações da rede metabólica, são rotulados com os números EC de tais enzimas. A figura 3.1 ilustra um exemplo de obtenção do grafo de enzimas.

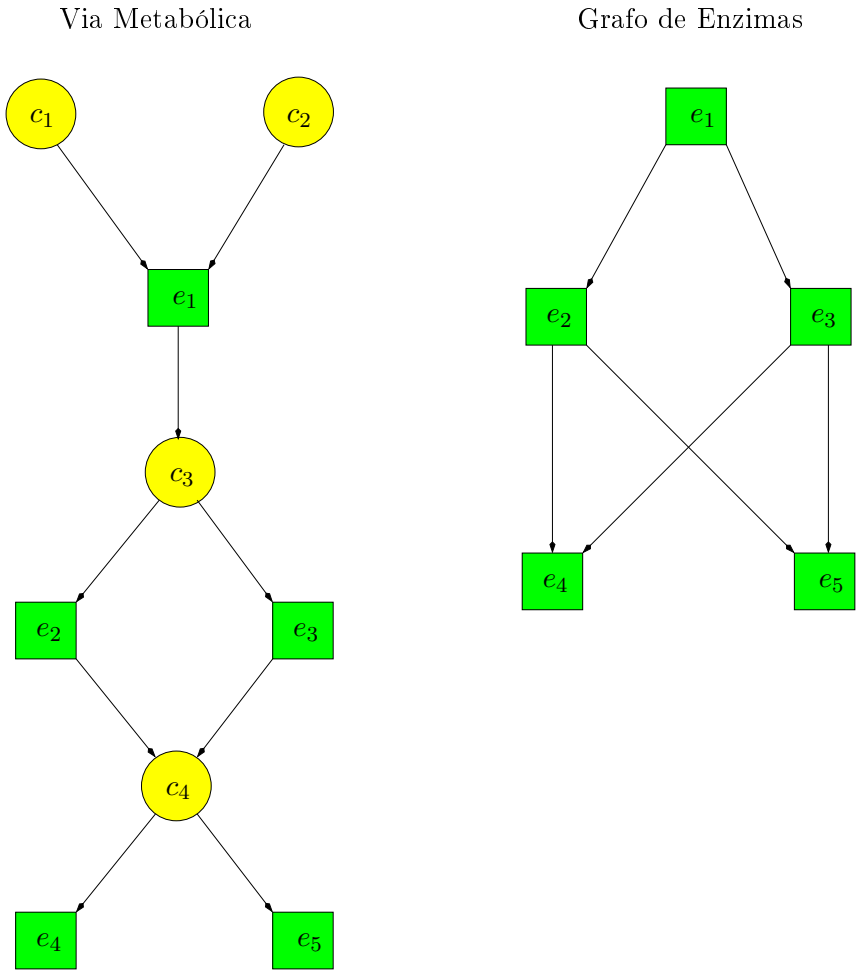

Figura 3.1 A partir de uma via metabólica um grafo de enzimas é obtido.

3.3 Obtenção da similaridade entre cada par de grafos de enzimas

Obtidos os grafos de enzimas, agora este método computa a similaridade entre cada par de grafos. O processo para computar a similaridade entre dois grafos de enzimas G_1 e G_2 é dividido em 3 fases. Na primeira, a similaridade entre cada par de vértices u, v, onde $u \in V_{G_1}$ e $v \in V_{G_2}$, é computada em um processo iterativo. Na segunda fase, um grafo bipartido é construído a partir das medidas de similaridade obtidas e o emparelhamento de custo máximo é obtido neste grafo. Na terceira fase, a medida de similaridade entre cada par de vértices saturados é recomputada em um processo iterativo e a similaridade entre os dois grafos é obtida pela soma das similaridades dos vértices saturados, normalizando essa soma.

Primeiramente, para dois grafos de enzimas G_1 e G_2, a **similaridade entre um par de vértices** $\{u, v\}$, onde $u \in V_{G_1}$ e $v \in V_{G_2}$, é dada pela comparação entre os números EC de cada vértice. Denotamos a similaridade entre um par de vértices $\{u, v\}$, onde $u \in V_{G_1}$ e $v \in V_{G_2}$, por $\gamma(u, v)$. Dizemos que dois vértices possuem similaridade 1 se todos os 4 dígitos de seus números EC forem iguais, $0,75$ se os 3 primeiros dígitos forem iguais, $0,5$ se os dois primeiros forem iguais, $0,25$ se apenas o primeiro for igual e 0 se todos os dígitos são diferentes [35].

Se a comparação entre os números EC de dois vértices u e v, onde $u \in V_{G_1}$ e $v \in V_{G_2}$, for igual a 0 esses vértices são ditos **não-similares**. A **similaridade entre dois grafos** G_1 e G_2 é dada pela similaridade entre cada par de vértices de G_1 e G_2. Dessa forma, podemos construir uma **matriz de similaridade (inicial) entre dois grafos** $\Gamma^{(0)}$, de dimensão $|V_{G_1}| \times |V_{G_2}|$, tal que

$$\Gamma^{(0)}(u, v) = \gamma(u, v)\cdot$$

para todo $u \in V_{G_1}$ e $v \in V_{G_2}$.

A similaridade $\Gamma^{(k+1)}(u, v)$, para $k \geq 0$, entre os vértices $u \in V_{G_1}$ e $v \in V_{G_2}$ é dada pela seguinte fórmula:

$$\Gamma^{(k+1)}(u, v) = \frac{\sum_{i=1}^{4} A_i^{(k)}(u, v) - \sum_{i=1}^{4} B_i^{(k)}(u, v)}{4} \cdot \gamma(u, v)\,,$$

para $k \geq 0$, onde os termos $A_i^{(k)}$ são tipos de similaridades entre esse par de vértices e os termos $B_i^{(k)}$ são tipos de não-similaridade.

Para computar o termo $A_1^{(k)}(u, v)$, consideramos todos os arcos $xu \in E_{G_1}$ e $yv \in E_{G_2}$. Dessa forma, A_1 refere-se à similaridade entre os vértices u e v considerando os arcos que entram em u em G_1 e os arcos que entram em v em G_2. Somamos a similaridade do par de vértices x, y, com $x \in V_{G_1}$ e $y \in V_{G_2}$, tais que $xu \in E_{G_1}$ e $yv \in E_{G_2}$ e normalizamos essa soma pelo produto dos arcos que entram em u e em v, ou seja, $\mathrm{d}^-_{G_1}(u) \cdot \mathrm{d}^-_{G_2}(v)$. Caso não existam arcos que entram em u e v, isto é, caso $\mathrm{d}^-_{G_1}(u) = \mathrm{d}^-_{G_2}(v) = 0$, então $A_1^{(k)}$ é normalizada pelo produto dos vértices de G_1 e G_2, ou seja, $|V_{G_1}| \cdot |V_{G_2}|$. Se u ou v é tal que

$\mathrm{d}^-(u) = 0$ (ou $\mathrm{d}^-(v) = 0$), então $A_1^{(k)} = 0$. Dessa forma, temos:

$$A_1^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{xu \in E_{G_1} \\ yv \in E_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{\mathrm{d}^-_{G_1}(u) \cdot \mathrm{d}^-_{G_2}(v)}\,, & \text{se } \mathrm{d}^-_{G_1}(u) \neq 0 \text{ e } \mathrm{d}^-_{G_2}(v) \neq 0\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \mathrm{d}^-_{G_1}(u) = \mathrm{d}^-_{G_2}(v) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

Para o termo $A_2^{(k)}(u,v)$, consideramos todos os arcos $ux \in E_{G_1}$ e $vy \in E_{G_2}$. Dessa forma, A_2 refere-se à similaridade entre os vértices u e v considerando os arcos que saem de u em G_1 e os arcos que saem de v em G_2. Somamos a similaridade do par de vértices x, y, com $x \in V_{G_1}$ e $y \in V_{G_2}$, tais que $ux \in E_{G_1}$ e $vy \in E_{G_2}$ e normalizamos essa soma pelo produto dos arcos que saem de u e de v, ou seja, $\mathrm{d}^+_{G_1}(u) \cdot \mathrm{d}^+_{G_2}(v)$. Caso não existam arcos que saem de u e v, isto é, caso $\mathrm{d}^+_{G_1}(u) = \mathrm{d}^+_{G_2}(v) = 0$, então $A_2^{(k)}$ é normalizada pelo produto dos vértices de G_1 e G_2, ou seja, $|V_{G_1}| \cdot |V_{G_2}|$. Se u ou v é tal que $\mathrm{d}^+_{G_1}(u) = 0$ (ou $\mathrm{d}^+_{G_2}(v) = 0$), então $A_2^{(k)} = 0$. Dessa forma, temos:

$$A_2^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{ux \in E_{G_1} \\ vy \in E_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{\mathrm{d}^+_{G_1}(u) \cdot \mathrm{d}^+_{G_2}(v)}\,, & \text{se } \mathrm{d}^+_{G_1}(u) \neq 0 \text{ e } \mathrm{d}^+_{G_2}(v) \neq 0\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \mathrm{d}^+_{G_1}(u) = \mathrm{d}^+_{G_2}(v) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

A figura 3.2 ilustra os termos A_1 e A_2.

Consideramos todos os arcos $xu \notin E_{G_1}$ e $yv \notin E_{G_2}$ no cálculo do termo $A_3^{(k)}(u,v)$. Dessa forma, A_3 refere-se à similaridade entre os vértices u e v considerando os arcos que não entram em u em G_1 e os arcos que não entram em v em G_2. Somamos a similaridade do par de vértices x, y, com $x \in V_{G_1}$ e $y \in V_{G_2}$, tais que $xu \notin E_{G_1}$ e $yv \notin E_{G_2}$ e normalizamos essa soma pelo produto dos arcos que não entram em u e em v, ou seja, $\tilde{\mathrm{d}}^-_{G_1}(u) \cdot \tilde{\mathrm{d}}^-_{G_2}(v)$. Caso não existam arcos que não entram em u e v, isto é, caso $\tilde{\mathrm{d}}^-_{G_1}(u) = \tilde{\mathrm{d}}^-_{G_2}(v) = 0$, então $A_3^{(k)}$ é normalizada pelo produto dos vértices de G_1 e G_2, ou seja, $|V_{G_1}| \cdot |V_{G_2}|$. Se u ou v é tal que $\tilde{\mathrm{d}}^-_{G_1}(u) = |V_{G_1}|$ (ou $\tilde{\mathrm{d}}^-_{G_2}(v) = |V_{G_2}|$), então $A_3^{(k)} = 0$. Dessa forma, temos:

$$A_3^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{xu \notin E_{G_1} \\ yv \notin E_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{\tilde{\mathrm{d}}^-_{G_1}(u) \cdot \tilde{\mathrm{d}}^-_{G_2}(v)}\,, & \text{se } \mathrm{d}^-_{G_1}(u) \neq |V_{G_1}| \text{ e } \mathrm{d}^-_{G_2}(v) \neq |V_{G_2}|\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \tilde{\mathrm{d}}^-_{G_1}(v) = \tilde{\mathrm{d}}^-_{G_2}(v) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

Com relação a $A_4^{(k)}(u,v)$, consideramos todos os arcos $ux \notin E_{G_1}$ e $vy \notin E_{G_2}$. Dessa forma, A_4 refere-se à similaridade entre os vértices u e v considerando os arcos que não saem de u em G_1 e os arcos que não

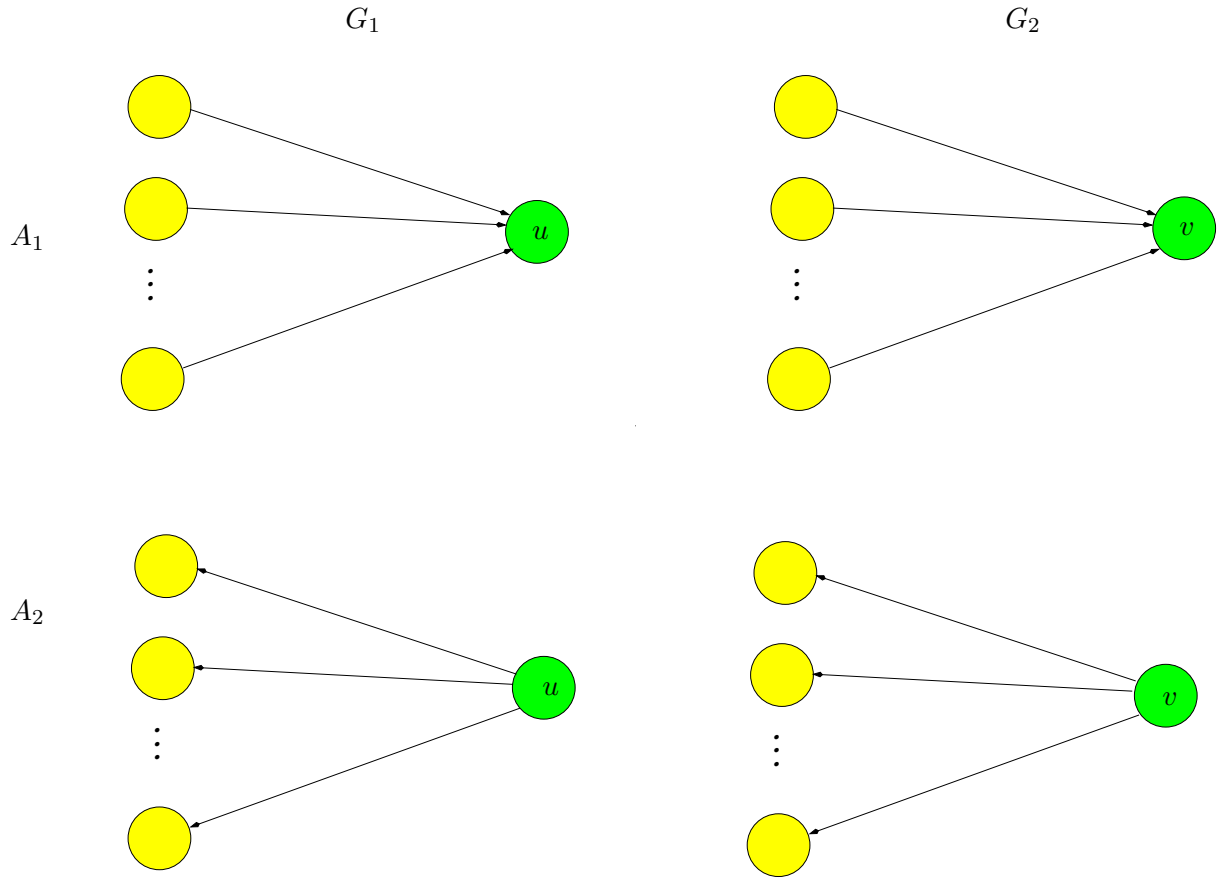

Figura 3.2 Termos A_1 e A_2.

saem de v em G_2. Somamos a similaridade do par de vértices x, y, com $x \in V_{G_1}$ e $y \in V_{G_2}$, tais que $ux \notin E_{G_1}$ e $vy \notin E_{G_2}$ e normalizamos essa soma produto dos arcos que não saem de u e não saem de v, ou seja, $\tilde{\mathrm{d}}^+_{G_1}(u) \cdot \tilde{\mathrm{d}}^+_{G_2}(v)$. Caso não existam arcos que não saem de u e v, isto é, caso $\tilde{\mathrm{d}}^+_{G_1}(u) = \tilde{\mathrm{d}}^+_{G_2}(v) = 0$, então $A_4^{(k)}$ é normalizada pelo produto dos vértices de G_1 e G_2, ou seja, $|V_{G_1}| \cdot |V_{G_2}|$. Se u ou v é tal que $\tilde{\mathrm{d}}^+_{G_1}(v) = |V_{G_1}|$ (ou $\tilde{\mathrm{d}}^+_{G_2}(v) = |V_{G_2}|$), então $A_4^{(k)} = 0$. Dessa forma, temos:

$$A_4^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{ux \notin E_{G_1} \\ vy \notin E_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{\tilde{\mathrm{d}}^+_{G_1}(u) \cdot \tilde{\mathrm{d}}^+_{G_2}(u)}\,, & \text{se } \mathrm{d}^+_{G_1}(u) \neq |V_{G_1}|\,,\ \text{e } \mathrm{d}^+_{G_2}(v) \neq |V_{G_2}| \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \tilde{\mathrm{d}}^+_{G_1}(u) = \tilde{\mathrm{d}}^+_{G_2}(u) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

A figura 3.3 ilustra os termos A_3 e A_4.

Em $B_1^{(k)}(u,v)$, consideramos todos os arcos $xu \in E_{G_1}$ e $yv \notin E_{G_2}$. Dessa forma, B_1 refere-se à similaridade entre os vértices u e v considerando os arcos que entram em u em G_1 e os arcos que não entram em v em G_2. Somamos a similaridade do par de vértices x, y, com $x \in V_{G_1}$ e $y \in V_{G_2}$, tais que

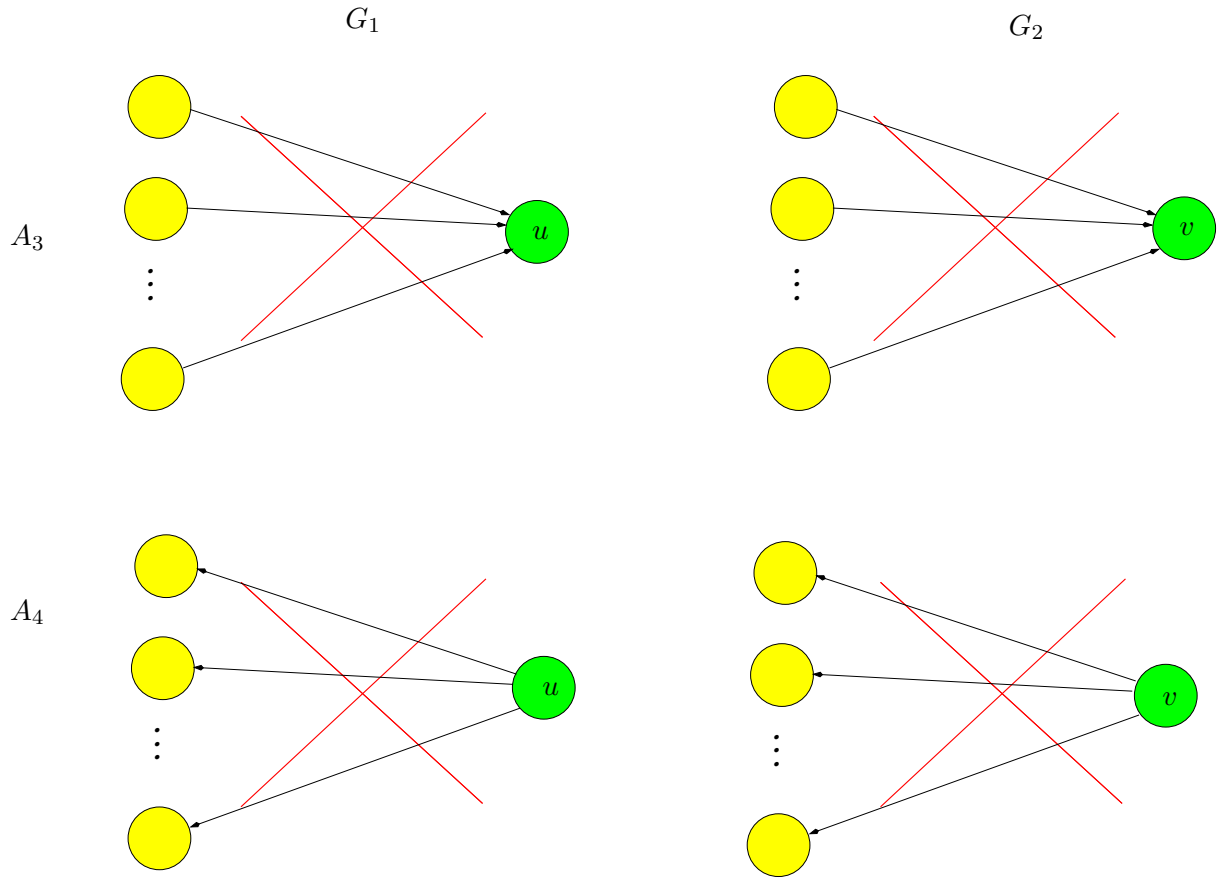

Figura 3.3 Termos A_3 e A_4.

$xu \in E_{G_1}$ e $yv \notin E_{G_2}$ e normalizamos essa soma pelo produto dos arcos que entram em u e não entram em v, ou seja, $\mathrm{d}^-_{G_1}(u) \cdot \tilde{\mathrm{d}}^-_{G_2}(v)$. Caso não existam arcos que entram em u e não existam arcos que não entram em v, isto é, caso $\mathrm{d}^-_{G_1}(u) = \tilde{\mathrm{d}}^-_{G_2}(v) = 0$, então $B_1^{(k)}$ é normalizada pelo produto dos vértices de G_1 e G_2, ou seja, $|V_{G_1}| \cdot |V_{G_2}|$. Se u ou v é tal que $\mathrm{d}^-_{G_1}(v) = 0$ (ou $\tilde{\mathrm{d}}^-_{G_2}(v) = |V_{G_2}|$), então $B_1^{(k)} = 0$. Dessa forma, temos:

$$B_1^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{xu \in E_{G_1} \\ yv \notin E_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{\mathrm{d}^-_{G_1}(u) \cdot \tilde{\mathrm{d}}^-_{G_2}(v)}\,, & \text{se } \mathrm{d}^-_{G_1}(u) \neq 0 \text{ e } \mathrm{d}^-_{G_2}(v) \neq |V_{G_2}|\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \mathrm{d}^-_{G_1}(u) = \tilde{\mathrm{d}}^-_{G_2}(v) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

Para $B_2^{(k)}(u,v)$, consideramos todos os arcos $xu \notin E_{G_1}$ e $yv \in E_{G_2}$. Dessa forma, B_2 refere-se à similaridade entre os vértices u e v considerando os arcos que não entram em u em G_1 e os arcos que entram em v em G_2. Somamos a similaridade do par de vértices x, y, com $x \in V_{G_1}$ e $y \in V_{G_2}$, tais que $xu \notin E_{G_1}$ e $yv \in E_{G_2}$ e normalizamos essa soma pelo produto dos arcos que não entram em u e entram

em v, ou seja, $\tilde{\mathrm{d}}^-_{G_1}(u) \cdot \mathrm{d}^-_{G_2}(v)$. Caso não existam arcos que não entram em u e não exitam arcos que entram em v, isto é, caso $\tilde{\mathrm{d}}^-_{G_1}(u) = \mathrm{d}^-_{G_2}(v) = 0$, então $B_2^{(k)}$ é normalizada pelo produto dos vértices de G_1 e G_2, ou seja, $|V_{G_1}| \cdot |V_{G_2}|$. Se u ou v é tal que $\tilde{\mathrm{d}}^-_{G_1}(v) = |V_{G_1}|$ (ou $\mathrm{d}^-_{G_2}(v) = 0$), então $B_2^{(k)} = 0$. Dessa forma, temos:

$$B_2^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{xu \notin E_{G_1} \\ yv \in E_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{\tilde{\mathrm{d}}^-_{G_1}(u) \cdot \mathrm{d}^-_{G_2}(v)}\,, & \text{se } \mathrm{d}^-_{G_1}(u) \neq |V_{G_1}| \text{ e } \mathrm{d}^-_{G_2}(v) \neq 0\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \tilde{\mathrm{d}}^-_{G_1}(u) = \mathrm{d}^-_{G_2}(v) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

No caso de $B_3^{(k)}(u,v)$, consideramos todos os arcos $ux \in E_{G_1}$ e $vy \notin E_{G_2}$. Dessa forma, B_3 refere-se à similaridade entre os vértices u e v considerando os arcos que saem de u em G_1 e os arcos que não saem de v em G_2. Somamos a similaridade do par de vértices x, y, com $x \in V_{G_1}$ e $y \in V_{G_2}$, tais que $ux \in E_{G_1}$ e $vy \notin E_{G_2}$ e normalizamos essa soma pelo produto dos arcos que entram em u e não entram em v, ou seja, $\mathrm{d}^+_{G_1}(u) \cdot \tilde{\mathrm{d}}^+_{G_2}(v)$. Caso não existam arcos que saem de u e não existam arcos que não saem de v, isto é, caso $\mathrm{d}^+_{G_1}(u) = \tilde{\mathrm{d}}^+_{G_2}(v) = 0$, então $B_3^{(k)}$ é normalizada pelo produto dos vértices de G_1 e G_2, ou seja, $|V_{G_1}| \cdot |V_{G_2}|$. Se u ou v é tal que $\mathrm{d}^+_{G_1}(v) = 0$ (ou $\tilde{\mathrm{d}}^+_{G_2}(v) = |V_{G_2}|$), então $B_3^{(k)} = 0$. Dessa forma, temos:

$$B_3^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{ux \in E_{G_1} \\ vy \notin E_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{\mathrm{d}^+_{G_1}(u) \cdot \tilde{\mathrm{d}}^+_{G_2}(v)}\,, & \text{se } \mathrm{d}^+_{G_1}(u) \neq 0 \text{ e } \mathrm{d}^+_{G_2}(v) \neq |V_{G_2}|\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \mathrm{d}^+_{G_1}(u) = \tilde{\mathrm{d}}^+_{G_2}(v) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

Por fim, para o termo $B_4^{(k)}(u,v)$, consideramos todos os arcos $ux \notin E_{G_1}$ e $vy \in E_{G_2}$. Dessa forma, B_4 refere-se à similaridade entre os vértices u e v considerando os arcos que não saem de u em G_1 e os arcos que saem de v em G_2. Somamos a similaridade do par de vértices x, y, com $x \in V_{G_1}$ e $y \in V_{G_2}$, tais que $ux \notin E_{G_1}$ e $vy \in E_{G_2}$ e normalizamos essa soma pelo produto dos arcos que não saem de u e saem de v, ou seja, $\tilde{\mathrm{d}}^+_{G_1}(u) \cdot \mathrm{d}^+_{G_2}(v)$. Caso não existam arcos que não saem de u e não exitam arcos que saem de v, isto é, caso $\tilde{\mathrm{d}}^+_{G_1}(u) = \mathrm{d}^+_{G_2}(v) = 0$, então $B_4^{(k)}$ é normalizada produto dos vértices de G_1 e G_2, ou seja, $|V_{G_1}| \cdot |V_{G_2}|$. Se u ou v é tal que $\tilde{\mathrm{d}}^+_{G_1}(v) = |V_{G_1}|$ (ou $\mathrm{d}^+_{G_2}(v) = 0$), então $B_4^{(k)} = 0$. Dessa forma, temos:

$$B_4^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{ux \notin E_{G_1} \\ vy \in E_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{\tilde{\mathrm{d}}^+_{G_1}(u) \cdot \mathrm{d}^+_{G_2}(v)}\,, & \text{se } \mathrm{d}^+_{G_1}(u) \neq |V_{G_1}| \text{ e } \mathrm{d}^+_{G_2}(v) \neq 0\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{\Gamma^{(k)}(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \tilde{\mathrm{d}}^+_{G_1}(u) = \mathrm{d}^+_{G_2}(v) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

A figura 3.4 ilustra os termos B_1 e B_3 e a figura 3.5 ilustra os termos B_2 e B_4.

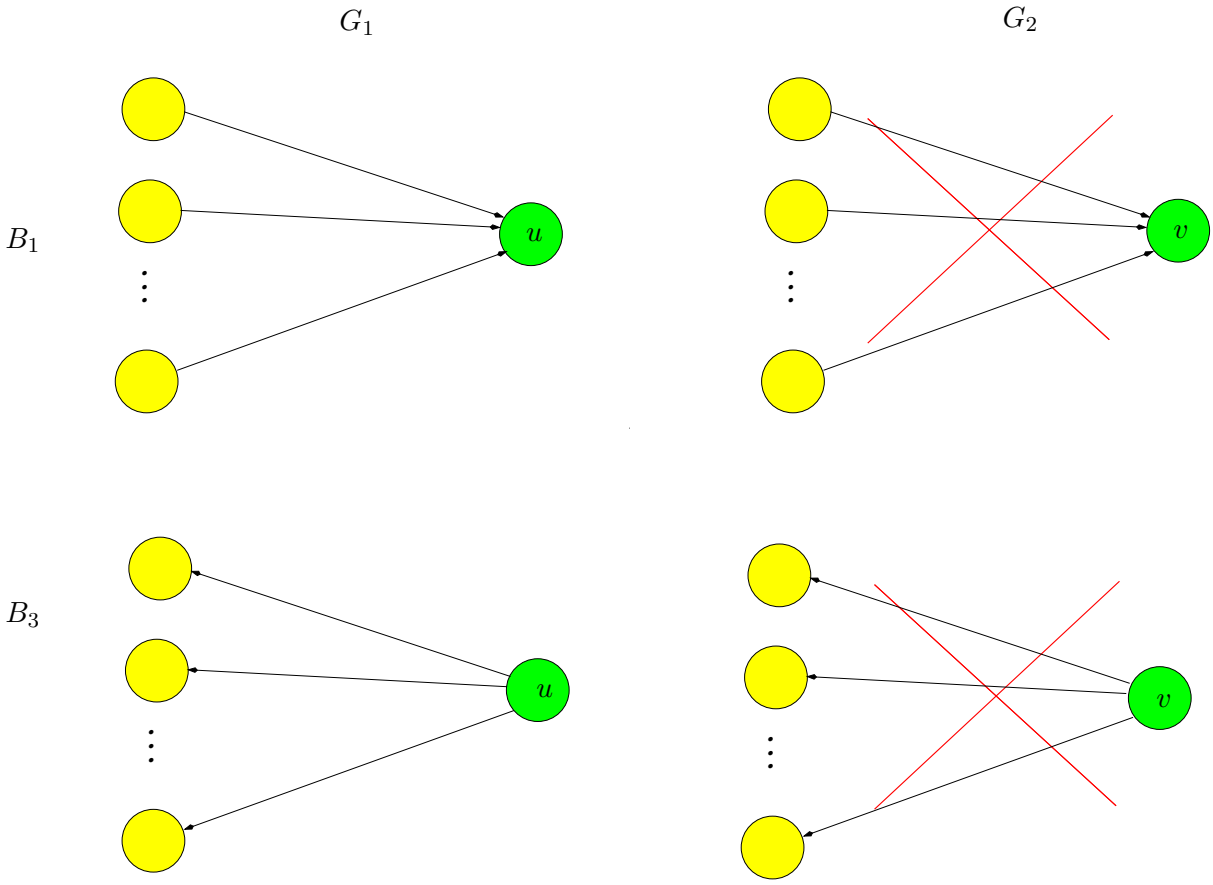

Figura 3.4 Termos B_1 e B_3.

Esse método converge para um valor $\Gamma^{(k^*)}(u,v)$, para todo $u \in V_{G_1}$ e $v \in V_{G_2}$, e para algum $k^* > 0$. A convergência é garantida em [35], já que as similaridades e não-similaridades entre vértices de dois grafos podem ser computadas como autovetores das matrizes correspondentes. Por fim, obtemos:

$$\Gamma^*(u,v) \leftarrow \frac{\Gamma^{(k^*)}(u,v)}{||\Gamma^{(k^*)}(u,v)||^2}\,,$$

para todo $u \in V_{G_1}$ e $v \in V_{G_2}$.

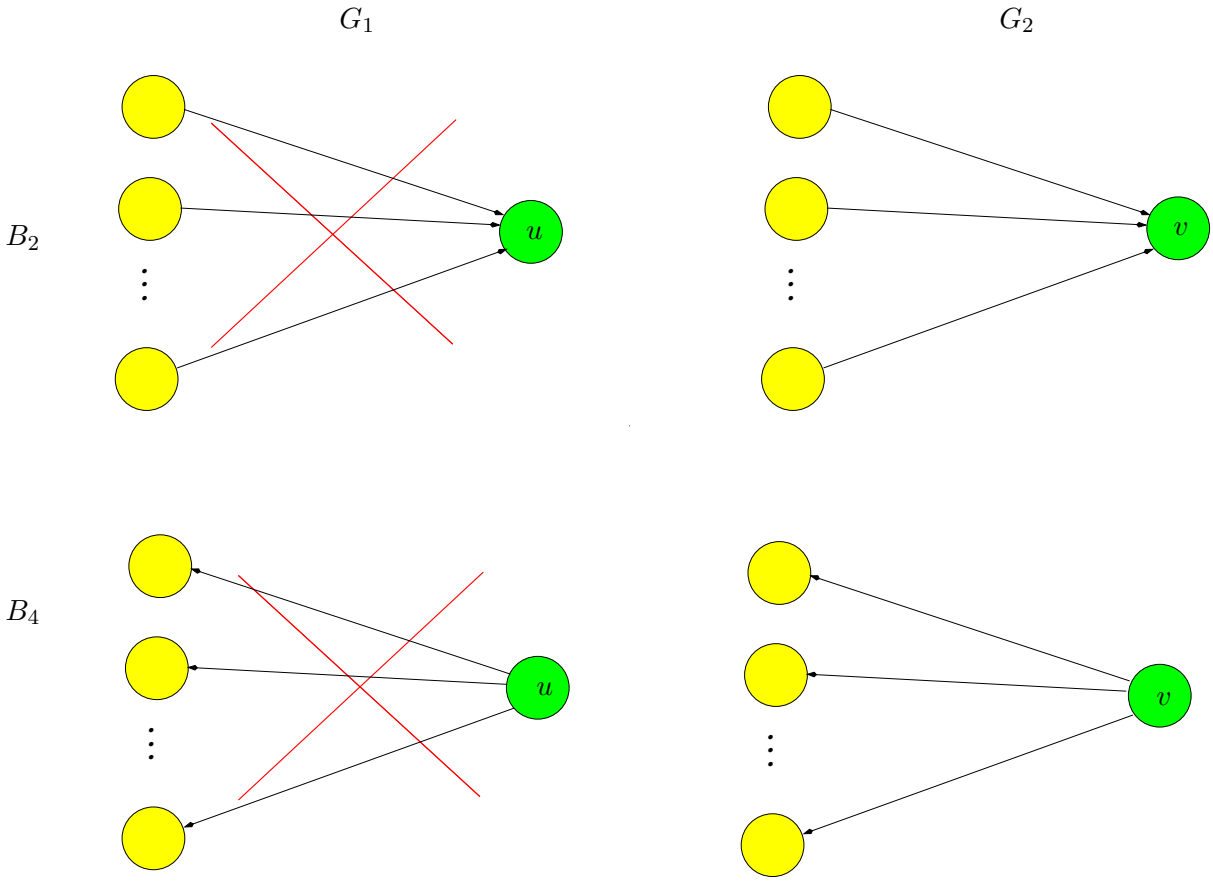

Figura 3.5 Termos B_2 e B_4.

Na segunda fase, a partir de Γ^*, encontramos um emparelhamento de custo máximo a partir dos grafos G_1 e G_2. Construimos então um grafo bipartido completo $H = (V, E)$, com $V_H = \{V_{G_1}, V_{G_2}, c\}$ (figura 3.6), e encontramos em H um emparelhamento de custo máximo $\mathcal{M}$, usando o algoritmo húngaro da seção 2.1.1, como pode ser visto na figura 3.7. Dizemos então que a matriz M, de dimensão $|V_{G_1}| \times |V_{G_2}|$, é tal que $M(u, v) = 1$ se u e v são vértices saturados pelo emparelhamento $\mathcal{M}$. Caso contrário, $M(u, v) = 0$.

Na terceira fase, usamos um processo semelhante ao da primeira fase para obter a similaridade $\Gamma_M(u, v)$ entre os vértices saturados pelo emparelhamento $\mathcal{M}$. Então, temos:

$$\Gamma_M(u, v) = \frac{\sum_{i=1}^{4} C_i(u, v) - \sum_{i=1}^{4} D_i(u, v)}{4} \cdot M(u, v) \, .$$

onde os termos C_i são tipos de similaridades e os termos D_i são tipos de não-similaridade entre esse par de vértices. O processo para computar os termos $C_1, \ldots, C_4$ e $D_1, \ldots, D_4$ é semelhante ao conjunto de equações anterior, exceto que na normalização da soma faremos a raiz quadrada da multiplicação dos graus de entrada/saída dos vértices e agora teremos apenas uma iteração com $M(x, y)$ em vez de $\gamma(x, y)$.

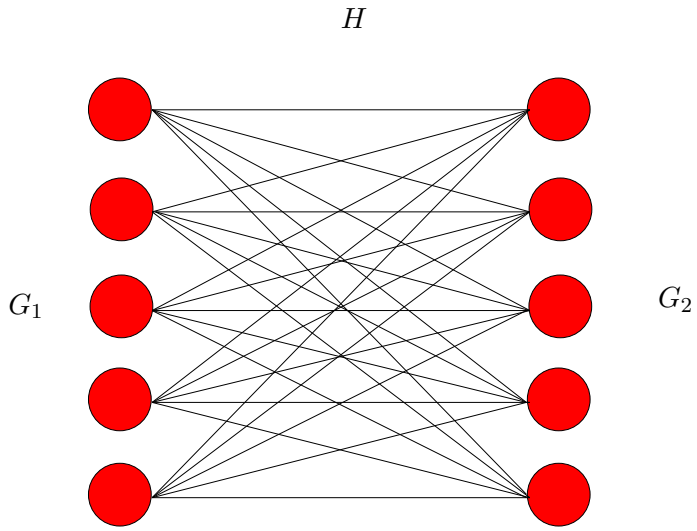

Figura 3.6 Grafo bipartido H.

Temos então:

$$C_1(u,v) = \begin{cases} \displaystyle\sum_{\substack{xu\in E_{G_1}\\ yv\in E_{G_2}}} \frac{M(x,y)}{\sqrt{\mathrm{d}^-_{G_1}(u)\cdot \mathrm{d}^-_{G_2}(v)}}\,, & \text{se } \mathrm{d}^-_{G_1}(u)\neq 0 \text{ e } \mathrm{d}^-_{G_2}(v)\neq 0\,,\\ \displaystyle\sum_{\substack{x\in V_{G_1}\\ y\in V_{G_2}}} \frac{M(x,y)}{|V_{G_1}|\cdot|V_{G_2}|}\,, & \text{se } \mathrm{d}^-_{G_1}(u)=\mathrm{d}^-_{G_2}(v)=0\,,\\ 0\,, & \text{caso contrário.}\end{cases}$$

$$C_2^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{ux\in E_{G_1}\\ vy\in E_{G_2}}} \frac{M(x,y)}{\sqrt{\mathrm{d}^+_{G_1}(u)\cdot \mathrm{d}^+_{G_2}(v)}}\,, & \text{se } \mathrm{d}^+_{G_1}(u)\neq 0 \text{ e } \mathrm{d}^+_{G_2}(v)\neq 0\,,\\ \displaystyle\sum_{\substack{x\in V_{G_1}\\ y\in V_{G_2}}} \frac{M(x,y)}{|V_{G_1}|\cdot|V_{G_2}|}\,, & \text{se } \mathrm{d}^+_{G_1}(u)=\mathrm{d}^+_{G_2}(v)=0\,,\\ 0\,, & \text{caso contrário.}\end{cases}$$

$$C_3^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{xu\notin E_{G_1}\\ yv\notin E_{G_2}}} \frac{M(x,y)}{\sqrt{\tilde{\mathrm{d}}^-_{G_1}(u)\cdot \tilde{\mathrm{d}}^-_{G_2}(v)}}\,, & \text{se } \mathrm{d}^-_{G_1}(u)\neq |V_{G_1}| \text{ e } \mathrm{d}^-_{G_2}(v)\neq |V_{G_2}|\,,\\ \displaystyle\sum_{\substack{x\in V_{G_1}\\ y\in V_{G_2}}} \frac{M(x,y)}{|V_{G_1}|\cdot|V_{G_2}|}\,, & \text{se } \tilde{\mathrm{d}}^-_{G_1}(v)=\tilde{\mathrm{d}}^-_{G_2}(v)=0\,,\\ 0\,, & \text{caso contrário.}\end{cases}$$

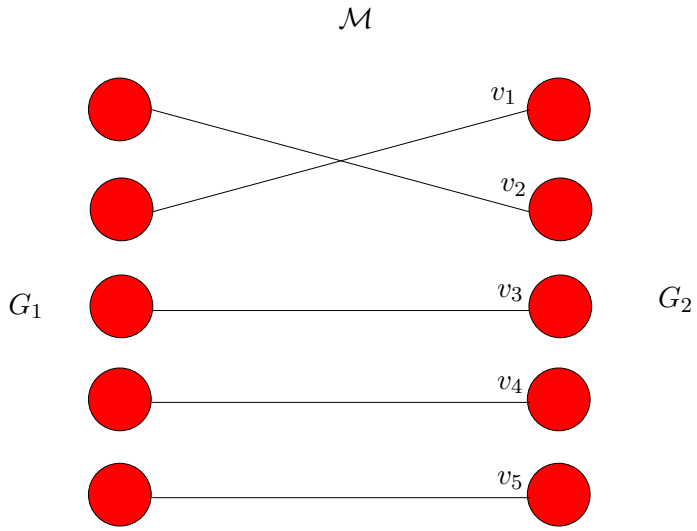

Figura 3.7 Emparelhamento de custo máximo $\mathcal{M}$.

$$C_4^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{ux \notin E_{G_1} \\ vy \notin E_{G_2}}} \frac{M(x,y)}{\sqrt{\tilde{\mathrm{d}}^+_{G_1}(u) \cdot \tilde{\mathrm{d}}^+_{G_2}(u)}}\,, & \text{se } \mathrm{d}^+_{G_1}(u) \neq |V_{G_1}| \text{ e } \mathrm{d}^+_{G_2}(v) \neq |V_{G_2}|\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{M(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \tilde{\mathrm{d}}^+_{G_1}(u) = \tilde{\mathrm{d}}^+_{G_2}(u) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

$$D_1^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{xu \in E_{G_1} \\ yv \notin E_{G_2}}} \frac{M(x,y)}{\sqrt{\mathrm{d}^-_{G_1}(u) \cdot \tilde{\mathrm{d}}^-_{G_2}(v)}}\,, & \text{se } \mathrm{d}^-_{G_1}(u) \neq 0 \text{ e } \mathrm{d}^-_{G_2}(v) \neq |V_{G_2}|\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{M(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \mathrm{d}^-_{G_1}(u) = \tilde{\mathrm{d}}^-_{G_2}(v) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

$$D_2^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{xu \notin E_{G_1} \\ yv \in E_{G_2}}} \frac{M(x,y)}{\sqrt{\tilde{\mathrm{d}}^-_{G_1}(u) \cdot \mathrm{d}^-_{G_2}(v)}}\,, & \text{se } \mathrm{d}^-_{G_1}(u) \neq |V_{G_1}| \text{ e } \mathrm{d}^-_{G_2}(v) \neq 0\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{M(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \tilde{\mathrm{d}}^-_{G_1}(u) = \mathrm{d}^-_{G_2}(v) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

$$D_3^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{ux \in E_{G_1} \\ vy \notin E_{G_2}}} \frac{M(x,y)}{\sqrt{\mathrm{d}^+_{G_1}(u) \cdot \tilde{\mathrm{d}}^+_{G_2}(v)}}\,, & \text{se } \mathrm{d}^+_{G_1}(u) \neq 0 \text{ e } \mathrm{d}^+_{G_2}(v) \neq |V_{G_2}|\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{M(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \mathrm{d}^+_{G_1}(u) = \tilde{\mathrm{d}}^+_{G_2}(v) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

$$D_4^{(k)}(u,v) = \begin{cases} \displaystyle\sum_{\substack{ux \notin E_{G_1} \\ vy \in E_{G_2}}} \frac{M(x,y)}{\sqrt{\tilde{\mathrm{d}}^+_{G_1}(u) \cdot \mathrm{d}^+_{G_2}(v)}}\,, & \text{se } \mathrm{d}^+_{G_1}(u) \neq |V_{G_1}| \text{ e } \mathrm{d}^+_{G_2}(v) \neq 0\,, \\ \displaystyle\sum_{\substack{x \in V_{G_1} \\ y \in V_{G_2}}} \frac{M(x,y)}{|V_{G_1}| \cdot |V_{G_2}|}\,, & \text{se } \tilde{\mathrm{d}}^+_{G_1}(u) = \mathrm{d}^+_{G_2}(v) = 0\,, \\ 0\,, & \text{caso contrário.} \end{cases}$$

Por fim, para obtermos a similaridade final Γ_{G_1,G_2} somamos os resultados computados na fase anterior e dividimos pela raiz quadrada da multiplicação do número de vértices de G_1 por G_2, ou seja, $\sqrt{|V_{G_1}| \cdot |V_{G_2}|}$. Temos então um valor entre -1 e 1 e quando $G_1 = G_2$ temos uma similaridade igual a 1.

$$\Gamma_{G_1,G_2} = \frac{\displaystyle\sum_{\substack{u \in V_{G_1}, v \in V_{G_2} \\ M(u,v)=1}} \Gamma_M(u,v)}{\sqrt{|V_{G_1}| \cdot |V_{G_2}|}}\,.$$

3.4 Construção da árvore filogenética

A partir da matriz de distâncias Γ obtida na seção anterior, uma árvore filogenética é construída usando um método baseado em distâncias chamado *Neighbor Joining* (seção 2.3.1), encontrado no pacote de programas *Phylip* [27], fornecendo como entrada a matriz Γ.

3.5 Experimentos e implementação

Implementamos o método de Heymans & Singh [35] usando a linguagem de programação C++ em um computador com processador Intel Core 2 Duo CPU E8300 2.83GHz e 4 GB de memória RAM. As entradas foram processadas a partir dos arquivos no formato XML obtidos no KEGG por um *analisador* implementado junto ao método, que extrai as informações de presença de enzimas e suas ligações dos arquivos e gera um grafo de enzimas que é a entrada para o método de Heymans & Singh [35]. O programa implementado recebe como entrada dois ou mais grafos de enzimas gerados pelos *analisador* e devolve como saída uma matriz de distâncias. Para entradas pequenas, com até 12 grafos, as execuções tiveram tempo de execução de uma fração de segundos.

Nome no KEGG	Organismo	Domínio	Categoria
Afu	*Archaeoglobus fulgidus*	Arqueia	Euryarchaeota
Ape	*Aeropyrum pernix*	Arqueia	Crenarchaeota
Hal	*Halobacterium sp. NRC-1*	Arqueia	Euryarchaeota
Mja	*Methanocaldococcus jannaschii*	Arqueia	Euryarchaeota
Mth	*Methanothermobacter thermautotrophicus*	Arqueia	Euryarchaeota
Pab	*Pyrococcus abyssi*	Arqueia	Euryarchaeota
Pfu	*Pyrococcus furiosus DSM 3638*	Arqueia	Euryarchaeota
Pho	*Pyrococcus horikoshii*	Arqueia	Euryarchaeota
Sso	*Sulfolobus solfataricus P2*	Arqueia	Crenarchaeota
Tac	*Thermoplasma acidophilum*	Arqueia	Euryarchaeota
Aae	*Aquifex aeolicus*	Bactéria	Hyperthermophilic bacteria
Bha	*Bacillus halodurans*	Bactéria	Firmicutes/Bacillales
Bsu	*Bacillus subtilis*	Bactéria	Firmicutes/Bacillales
Dra	*Deinococcus radiodurans*	Bactéria	Deinococcus-Thermus
Lla	*Lactococcus lactis*	Bactéria	Firmicutes/Lactobacillales
Mle	*Mycobacterium leprae*	Bactéria	Actinobacteria
Mtu	*Mycobacterium tuberculosis H37Rv*	Bactéria	Actinobacteria
Sau	*Staphylococcus aureus N315*	Bactéria	Firmicutes/Bacillales
Sav	*Staphylococcus aureus Mu50*	Bactéria	Firmicutes/Bacillales
Spy	*Streptococcus pyogenes SF370*	Bactéria	Firmicutes/Lactobacillales
Tma	*Thermotoga maritima*	Bactéria	Hyperthermophilic bacteria
Xfa	*Xylella fastidiosa 9a5c*	Bactéria	Proteobacteria/Gamma/Others

Tabela 3.1 *Tabela dos 22 organismos selecionados para o experimento.*

Realizamos um pequeno experimento usando essa implementação do algoritmo de Heymans & Singh [35]. Inicialmente, selecionamos 22 organismos presentes no KEGG divididos em 2 conjuntos: um com 10 organismos do domínio Arqueia e outro com 12 organismos do domínio Bactéria. A tabela 3.1 apresenta esses organismos selecionados. Buscando evitar a seleção de organismos aleatórios, escolhemos organismos de um outro experimento que resultou na árvore da figura 4.4, obtida por Zhang et al. [63] e apresentada no capítulo 4.

A partir desses dois conjuntos, executamos o método de Heymans & Singh [35] para o metabolismo da Glicólise. A rede metabólica da Glicólise foi escolhida por estar presente em praticamente todos os organismos vivos. Além disso, ela foi uma das primeiras redes metabólicas estudadas e é uma das mais bem entendidas, em termos das enzimas envolvidas no metabolismo [35].

Para fins de comparação, construímos árvores baseadas em cadeias de 16S rRNA através dos dados obtidos no repositório do *Ribosomal Database Project-II* [18]. Uma cadeia de RNA ribossomal 16S, ou simplesmente 16S rRNA, é um componente da subunidade 30S dos ribossomos de procariotos, com 1542 nucleotídeos. Os genes que a codificam são chamados de 16S rRNA e são usados na construção de árvores filogenéticas, já que são altamente conservados entre as espécies de bactérias e arqueias. O repositório permite que façamos escolhas de conjuntos de organismos e nos fornece matrizes de distâncias a partir das bases de 16S rRNA dos organismos. Com essas matrizes construímos árvores filogenéticas usando o método baseado em distâncias *Neighbor Joining* apresentado na seção 2.3.1.

3.5.1 Árvores filogenéticas obtidas

A tabela 3.2 foi obtida pela execução do método de Heymans & Singh [35] para o conjunto de 10 organismos do domínio Arqueia. A figura 3.8(a) mostra a árvore filogenética construída através dessa

	Afu	Ape	Hal	Mja	Mth	Pab	Pfu	Pho	Sso	Tac
Afu	0.00	0.70	0.71	0.52	0.48	0.52	0.62	0.61	0.65	0.73
Ape	0.70	0.00	0.25	0.61	0.93	0.49	0.46	0.47	0.31	0.39
Hal	0.71	0.25	0.00	0.53	0.56	0.48	0.49	0.46	0.31	0.39
Mja	0.52	0.61	0.53	0.00	0.63	0.49	0.60	0.57	0.65	0.72
Mth	0.48	0.93	0.56	0.63	0.00	0.92	0.72	0.72	0.78	0.95
Pab	0.52	0.49	0.48	0.49	0.92	0.00	0.36	0.23	0.48	0.54
Pfu	0.62	0.46	0.49	0.60	0.72	0.36	0.00	0.22	0.41	0.44
Pho	0.61	0.47	0.46	0.57	0.72	0.23	0.22	0.00	0.45	0.51
Sso	0.65	0.31	0.31	0.65	0.78	0.48	0.41	0.45	0.00	0.30
Tac	0.73	0.39	0.39	0.72	0.95	0.54	0.44	0.51	0.30	0.00

Tabela 3.2 *Matriz de distâncias obtida pelo método de Heymans & Singh [35] para o conjunto de 10 organismos do domínio Arqueia.*

	Ape	Sso	Afu	Hal	Mth	Mja	Pab	Pho	Pfu	Tac
Ape	0.0	0.12	0.19	0.25	0.19	0.17	0.15	0.15	0.15	0.27
Sso	0.12	0.00	0.22	0.28	0.22	0.19	0.21	0.21	0.21	0.29
Afu	0.19	0.22	0.00	0.23	0.16	0.14	0.14	0.14	0.14	0.23
Hal	0.25	0.28	0.23	0.00	0.21	0.23	0.23	0.22	0.22	0.26
Mth	0.19	0.22	0.16	0.21	0.00	0.16	0.16	0.16	0.16	0.23
Mja	0.17	0.19	0.14	0.23	0.16	0.00	0.13	0.12	0.12	0.23
Pab	0.15	0.21	0.14	0.23	0.16	0.13	0.00	0.01	0.01	0.23
Pho	0.15	0.21	0.14	0.22	0.16	0.12	0.01	0.00	0.00	0.23
Pfu	0.15	0.21	0.14	0.22	0.16	0.12	0.01	0.00	0.00	0.23
Tac	0.27	0.29	0.23	0.26	0.23	0.23	0.23	0.23	0.23	0.00

Tabela 3.3 *Matriz obtida pela distância entre as cadeias de 16S rRNA para o conjunto de 10 organismos do domínio Arqueia e encontrada no repositório do Ribosomal Database Project-II [18].*

matriz usando o *Neighbor Joining*. A figura 3.8(b) mostra a árvore filogenética dos mesmos 10 organismos, mas agora obtida pela tabela 3.3 que representa a distância das cadeias de 16S rRNA desses organismos, também usando o *Neighbor Joining*.

Como esperado, as árvores das figuras 3.8(a) e 4.4 apresentam similaridades em suas ramificações, como no agrupamento dos organismos *Pyrococcus abyssi*, *Pyrococcus horikoshii* e *Pyrococcus furiosus DSM 3638*. No entanto, essas árvores são muito diferentes da árvore da figura 3.8(b), como podemos perceber pelo agrupamento dos organismos *Archaeoglobus fulgidus*, *Halobacterium sp. NRC-1*,*Thermoplasma acidophilum* e *Mycobacterium tuberculosis H37Rv*. O ancestral comum a esses organismos não é um ancestral do organismo *Methanocaldococcus jannaschii*, como nas árvores das figuras 3.8(a) e 4.4.

A figura 3.9(a) mostra a árvore construída a partir da tabela 3.4 usando o *Neighbor Joining*, tomando a tabela 3.4 como entrada. Esta árvore foi obtida pela execução do método de Heymans & Singh [35] para o conjunto de 12 organismos do domínio Bactéria. A figura 3.9(b) mostra a árvore filogenética dos mesmos 12 organismos, representados por suas cadeias de 16S rRNA, obtida através da tabela 3.5, usando o *Neighbor Joining*.

As árvores 3.9(a) e 4.4 também apresentam similaridades em suas ramificações, como no agrupamento dos organismos *Lactococcus lactis* , *Streptococcus pyogenes SF370* e também dos organismos *Staphylococcus aureus N315*, *Staphylococcus aureus Mu50*. No entanto, a árvore 3.9(b) também apresenta esses mesmos agrupamentos e mais o dos organismos *Bacillus halodurans* e *Bacillus subtilis*.

	Aae	Bha	Bsu	Dra	Lla	Mle	Mtu	Sau	Sav	Spy	Tma	Xfa
Aae	0.00	0.76	0.69	0.64	0.62	0.44	0.56	0.68	0.68	0.68	0.60	0.71
Bha	0.76	0.00	0.31	0.44	0.42	0.75	0.42	0.25	0.25	0.46	0.59	0.47
Bsu	0.69	0.31	0.00	0.39	0.36	0.45	0.43	0.22	0.22	0.38	0.54	0.41
Dra	0.64	0.44	0.39	0.00	0.39	0.42	0.16	0.39	0.39	0.44	0.55	0.45
Lla	0.62	0.42	0.36	0.39	0.00	0.50	0.46	0.46	0.46	0.33	0.36	0.38
Mle	0.44	0.75	0.45	0.42	0.50	0.00	0.48	0.56	0.56	0.56	0.47	0.47
Mtu	0.56	0.42	0.43	0.16	0.46	0.48	0.00	0.44	0.44	0.50	0.54	0.45
Sau	0.68	0.25	0.22	0.39	0.46	0.56	0.44	0.00	0.00	0.43	0.54	0.50
Sav	0.68	0.25	0.22	0.39	0.46	0.56	0.44	0.00	0.00	0.43	0.54	0.50
Spy	0.68	0.46	0.38	0.44	0.33	0.56	0.50	0.43	0.43	0.00	0.45	0.40
Tma	0.60	0.59	0.54	0.55	0.36	0.47	0.54	0.54	0.54	0.45	0.00	0.44
Xfa	0.71	0.47	0.41	0.45	0.38	0.47	0.45	0.50	0.50	0.40	0.44	0.00

Tabela 3.4 *Matriz de distâncias obtida pelo método de Heymans & Singh [35] para o conjunto de 12 organismos do domínio Bactéria.*

	Mtu	Mle	Aee	Dra	Xfa	Tma	Bsu	Bha	Sau	Sav	Spy	Lla
Mtu	0.00	0.01	0.23	0.22	0.19	0.20	0.18	0.18	0.19	0.19	0.21	0.20
Mle	0.01	0.00	0.23	0.22	0.20	0.20	0.18	0.18	0.19	0.19	0.21	0.20
Aee	0.23	0.23	0.00	0.27	0.25	0.18	0.23	0.23	0.24	0.24	0.25	0.24
Dra	0.22	0.22	0.27	0.00	0.21	0.23	0.21	0.21	0.21	0.21	0.22	0.23
Xfa	0.19	0.20	0.25	0.21	0.00	0.21	0.18	0.18	0.18	0.18	0.19	0.19
Tma	0.20	0.20	0.18	0.23	0.21	0.00	0.20	0.20	0.21	0.21	0.21	0.21
Bsu	0.18	0.18	0.23	0.21	0.18	0.20	0.00	0.04	0.06	0.06	0.12	0.12
Bha	0.18	0.18	0.23	0.21	0.18	0.20	0.04	0.00	0.07	0.07	0.11	0.11
Sau	0.19	0.19	0.24	0.21	0.18	0.21	0.06	0.07	0.00	0.00	0.12	0.13
Sav	0.19	0.19	0.24	0.21	0.18	0.21	0.06	0.07	0.00	0.00	0.12	0.13
Spy	0.21	0.21	0.25	0.22	0.19	0.21	0.12	0.11	0.12	0.12	0.00	0.08
Lla	0.20	0.20	0.24	0.23	0.19	0.21	0.12	0.11	0.13	0.13	0.08	0.00

Tabela 3.5 *Matriz obtida pela distância entre as cadeias de 16S rRNA para um conjunto de 12 organismos do domínio Bactéria e encontrada no repositório do Ribosomal Database Project-II [18].*

(a) **(b)**

Figura 3.8 Em (a) temos a árvore filogenética obtida pelo método de Heymans & Singh [35], a partir da tabela 3.2 e em (b) a árvore filogenética baseada na distância entre as cadeias de 16S rRNA para o conjunto de 10 organismos do domínio Arqueia, obtida através da tabela 3.3.

Comparar árvores filogenéticas é uma tarefa árdua pela não existência de um consenso sobre a corretude das árvores e das técnicas para construção exigindo um trabalho em conjunto com biólogos. Neste trabalho escolhemos comparar nossos experimentos com uma árvore já existente, da figura 4.4, pois não dispomos da ajuda de biólogos e seguindo por esse caminho de observações não conseguimos concluir se nosso método é melhor/mais eficiente do que o da árvore que nos baseamos.

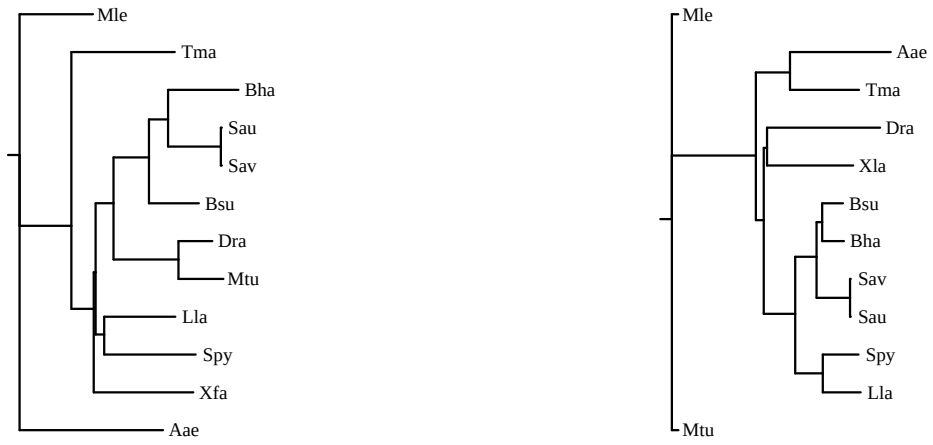

Figura 3.9 Em (a) temos a árvore filogenética obtida pelo método de Heymans & Singh [35], a partir da tabela 3.4 e em (b) a Árvore filogenética baseada na distância entre as cadeias de 16S rRNA para um conjunto de 12 organismos do domínio Bactéria, obtida a partir da tabela 3.5.

Capítulo 4

Método de Zhang et al. (2006)

O objetivo do método de Zhang *et al.* (2006) [63] é construir árvores filogenéticas de organismos considerando, além das relações evolutivas entre organismos – como usual em um método de construção de árvores filogenéticas –, também a interação entre a expressão dos seus genes e a influência de fatores ambientais. Para tanto, considera a estrutura das vias metabólicas do organismo, que pode ser vista como resultante do compromisso entre a informação filogenética herdada pelo último ancestral comum e a pressão evolutiva que provoca ajustes na estrutura metabólica da via.

4.1 Algoritmo

O método de Zhang *et al.*(2006) [63] pode ser apresentado como um algoritmo que recebe vias metabólicas de um conjunto de organismos e devolve como saída uma árvore filogenética desses organismos.

Algoritmo Zhang et al. (2006)

Entrada: recebe um conjunto de organismos $\mathcal{O}$ e o conjunto de todas vias metabólicas $\mathcal{P}$ dos organismos
Saída: devolve uma árvore filogenética gerada a partir do conjunto de organismos

1: obtenha um grafo via para cada via metabólica $p_k \in \mathcal{P}$
2: obtenha um grafo base H_k^o para cada via metabólica p_k em cada organismo $o \in \mathcal{O}$
3: obtenha o conjunto O_k de organismos em que p_k está presente
4: compute uma matriz de distâncias D_k a partir de cada conjunto O_k
5: construa uma árvore filogenética T_k, a partir de cada matriz D_k
6: construa um conjunto de quartetos Q_k a partir de O_k, p_k, D_k e T_k. Combine os conjuntos Q_k em um único conjunto de quartetos $\mathcal{Q} = \cup Q_k$ e construa a árvore filogenética T^* a partir de $\mathcal{Q}$ usando o método dos quartetos Q^*

4.2 Obtenção do grafo via

Seja $\mathcal{P} = \{p_1, \ldots, p_n\}$ um conjunto de n vias metabólicas. Para cada via metabólica p em $\mathcal{P}$, um grafo orientado G é construído, tal que seus vértices representam as enzimas constantes em p e os arcos $\vec{uv}$ indicam que um produto da reação catalisada pela enzima u é usado como substrato da reação catalisada

por v. Chame o grafo G de **grafo via**. Sejam $G_1, \ldots, G_n$ os grafos via assim construídos. A figura 3.1, mostra a obtenção de um grafo via, que é similar a obtenção do grafo de enzimas visto no capítulo 3.

4.3 Obtenção do grafo base

Seja $\mathcal{O}$ um conjunto de organismos. Para cada organismo o em $\mathcal{O}$, seja Z^o o conjunto de enzimas presentes em o. Para cada organismo o, os grafos orientados $H_1^o, \ldots, H_m^o$ são construídos de tal forma que $H_k^o = G_k[Z^o]$, isto é, H_k^o é um subgrafo gerador do grafo via G_k com conjunto de vértices Z^o. Considere apenas os grafos H_k^o não-triviais. Assim, $m \leq n$. Chame H_k^o de **grafo base da via metabólica p_k no organismo o** ou, quando o contexto permitir, simplesmente **grafo base**. A figura 4.1 ilustra o processo de obtenção de um grafo base.

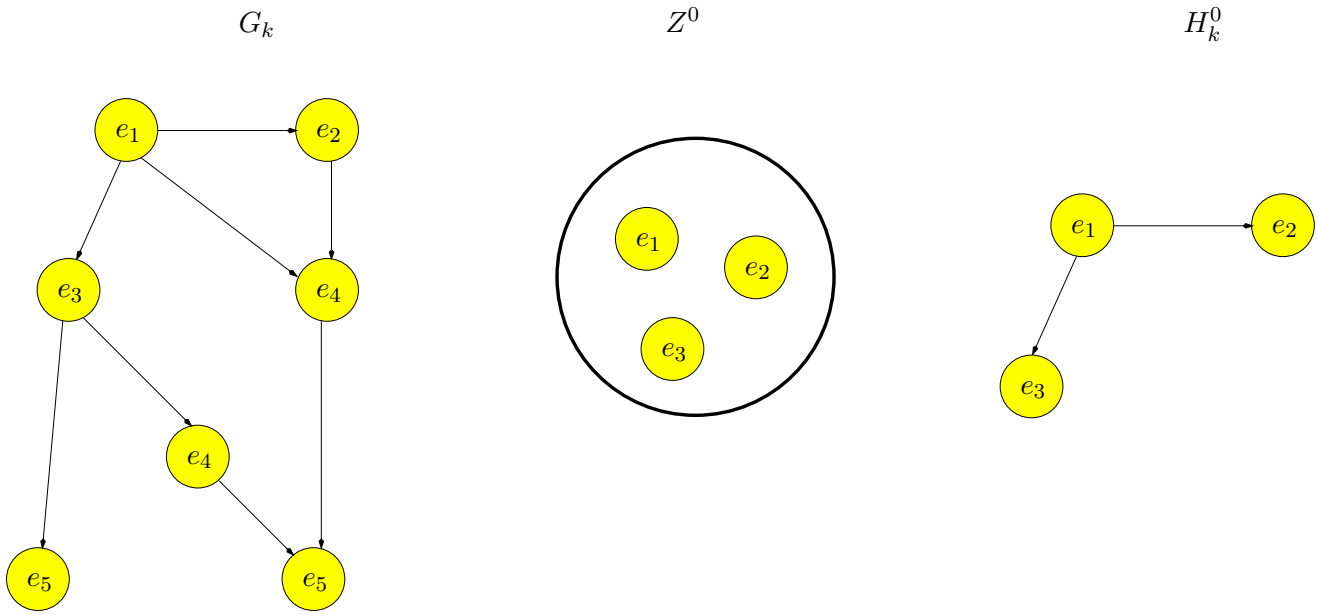

Figura 4.1 Processo de obtenção do grafo base.

4.4 Conjunto de organismos O_k que contém uma via metabólica p_k

Uma via metabólica p_k é dita **presente** em um organismo o se $\text{diam}(H_k^o) > \mu(G_k)$, onde $\text{diam}(H_k^o)$ denota o diâmetro de H_k^o, isto é, o maior comprimento dos menores caminhos entre todos os pares de vértices de H_k^o, e $\mu(G_k)$ denota o comprimento médio dos menores caminhos entre todos os pares de vértices de G_k. Caso contrário, a via metabólica p_k é considerada **ausente** no organismo o.

Considere O_k o conjunto dos organismos em $\mathcal{O}$ que têm presente a via metabólica p_k, como ilustrado na figura 4.2. Sejam H_k^o e H_k^r os grafos base de p_k nos organismos o e r de O_k. A distância entre H_k^o e

H_k^r, definida por Zhang *et al.* em [63], é dada por

$$\text{dist}(H_k^o, H_k^r) = 1 - \frac{\displaystyle\sum_{u \in V_{H_k^o} \cap V_{H_k^r}} \frac{2|N_{H_k^o}(u) \cap N_{H_k^r}(u)|}{|N_{H_k^o}(u)| + |N_{H_k^r}(u)|}}{\sqrt{|V_{H_k^o}| \cdot |V_{H_k^r}|}} \ ,$$

onde $N_G(u)$ representa o conjunto de vizinhos do vértice u no grafo G.

O_k

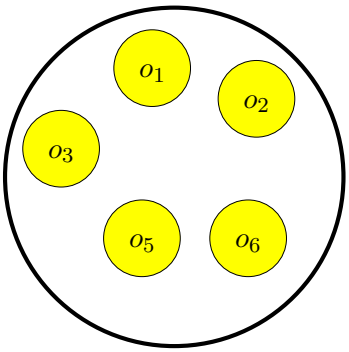

Figura 4.2 Conjunto de organismos O_k.

4.5 Árvores filogenéticas T_k e árvore filogenética final

Para cada via metabólica p_k, uma matriz de distâncias D_k é construída onde um elemento $d_{o,r}^k$ da matriz D_k representa a distância $\text{dist}(H_k^o, H_k^r)$ entre os grafos base H_k^o e H_k^r nos organismos o e r em O_k, isto é, nos quais a via p_k está presente. A partir da matriz D_k uma árvore filogenética T_k é construída para cada via metabólica p_k, usando um método de construção de árvores filogenéticas baseado em distâncias (seção 2.3.1). A figura 4.3 exemplifica uma árvore filogenética T_k.

A partir do conjunto de organismos O_k associado à via metabólica p_k, da matriz de distâncias D_k e da árvore T_k, um conjunto de quartetos Q_k é construído (seção 2.3.2). Em seguida, os conjuntos Q_k são combinados em um único conjunto de quartetos $\mathcal{Q} = \cup Q_k$. Por fim, a partir de $\mathcal{Q}$ é construída uma árvore filogenética T^* baseada no conjunto completo de vias metabólicas presentes nos organismos, usando o método dos quartetos Q^* [9]. A figura 4.4 mostra a árvore filogenética obtida no fim do processo do método de Zhang *et al.* (2006) [63].

4.6 Considerações biológicas e experimentos

No trabalho de Zhang *et al.* em [63], um total de 103 vias metabólicas foram extraídas do repositório KEGG PATHWAY [38] e transformadas em grafos via $G_1, \ldots, G_{103}$.

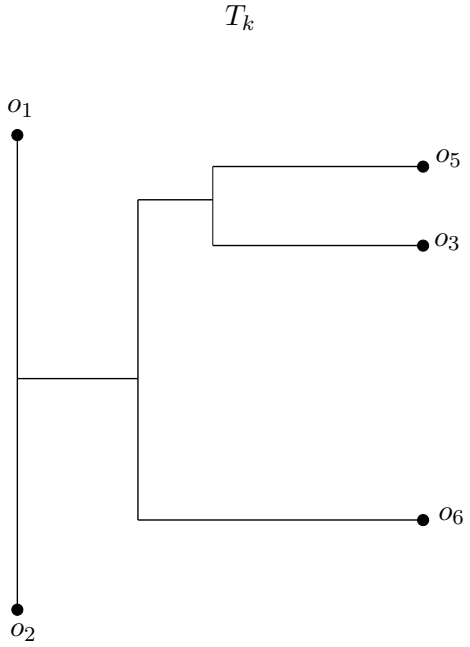

Figura 4.3 Exemplo de árvore filogenética T_k obtida a partir de uma matriz D_k.

Um total de 184 organismos completamente sequenciados foram usados no experimento, incluindo 19 arqueias, 152 bactérias e 13 eucárias, todos anotados no KEGG GENE [38], com a classificação dos mesmos em 3 domínios e 27 categorias. A informação da presença de uma enzima em um organismo foi obtida da seção ENZYME do KEGG LIGAND [38]. A partir dessas informações, foram construídos os grafos base para cada via metabólica em todos os organismos estudados.

A definição da presença de uma via metabólica em um organismo apresentada no trabalho de Zhang *et al.* em [63], baseada no diâmetro do grafo base e no comprimento médio dos menores caminhos do grafo via, é justificada pela observação que o grafo base da via no organismo deve conter pelo menos uma sequência contínua de reações químicas relativamente extensa se comparada com o "tamanho" do grafo via.

Dada a definição de presença ou ausência de vias metabólicas em organismos de Zhang *et al.* em [63], para cada via metabólica existe um subconjunto de organismos que a contém. Em geral, os organismos em um subconjunto de organismos associado a uma via metabólica não são igualmente distribuídos entre as diferentes categorias filogenéticas. Dessa forma, o método atribui valores-P para distinguir categorias que tendem a ter mais ou menos organismos em um subconjunto de organismos associado a uma via metabólica. O valor-P^+ é a probabilidade de se observar, ao acaso, pelo menos k organismos em uma subconjunto de n organismos associado a uma via metabólica, todos eles pertencendo a uma categoria contendo C organismos, de um total de O organismos:

$$\mathrm{P}^+ = 1 - \sum_{i=0}^{k-1} \frac{\binom{C}{i}\binom{O-C}{n-i}}{\binom{O}{n}}.$$

O valor-P^+ fornece uma medida para verificar se um subconjunto de organimos associado a uma via metabólica tem mais organismos de uma categoria particular do que o esperado ao acaso. Por outro lado, o valor-P^- é a probabilidade de se observar, ao acaso, não mais que k organismos em uma subconjunto

de n organismos associado a uma via metabólica, todos eles pertencendo a uma categoria contendo C organismos, de um total de O organismos:

$$\mathrm{P}^- = \sum_{i=0}^{k-1} \frac{\binom{C}{i}\binom{O-C}{n-i}}{\binom{O}{n}} .$$

O valor-P^- fornece uma medida para verificar se um subconjunto de organimos associado a uma via metabólica tem menos organismos de uma categoria particular do que o esperado ao acaso.

A distância entre dois grafos base de uma via metabólica em dois organismos, usada no trabalho de Zhang *et al.* em [63], é baseada nas relações estruturais entre vértices compartilhados nesses grafos. Pelos mesmos motivos, a distância definida por Heymans e Singh em [35] também pode ser usada neste método.

A construção das árvores filogenéticas baseadas nos grafos base das vias metabólicas nos organismos é realizada em [63] selecionando, do conjunto inicial de organismos, apenas os procariotos. O programa NEIGHBOR do pacote PHYLIP [27], baseado no algoritmo *Neighbor Joining* da seção 2.3.1, foi usado para construir, a partir das matrizes de distâncias D_k, as árvores filogenéticas T_k associadas à O_k.

Finalmente, o método dos quartetos foi usado para construção de uma árvore filogenética única T^* baseada em todas as árvore filogenéticas T_k associadas à O_k. No experimento realizado em [63] foram considerados 47 organismos. Para cada subconjunto de organismos O_k associado a uma via metabólica p_k com pelo menos 4 organismos, com entradas no servidor SHOT [40], uma matriz de distâncias foi obtida, computando as distâncias entre pares de organismos o e r, representados por seus grafos base H_k^o e H_k^r. Em seguida, um conjunto de quartetos Q_k foi construído a partir da matriz de distâncias D_k usando o programa DISTQUART do pacote PHYLOQUART [9]. Os conjuntos Q_k foram então processados e combinados em um único conjunto de quartetos $\mathcal{Q} = \cup Q_k$ e, usando o método Q^* [9] do pacote PHYLOQUART, um conjunto de partições foi obtido e fornecido como entrada para o programa TREEPOP do pacote PHYLOQUART [9] que, finalmente, devolveu a árvore filogenética T^* baseada nas vias metabólicas dos organismos. A figura 4.4 mostra a árvore filogenética T^* baseada nas vias metabólicas dos 47 organismos do experimento realizado no trabalho de Zhang *et al.* [63].

Além das árvores filogenéticas baseadas nas vias metabólicas dos organismos, árvores baseadas na molécula 16S rRNA desses organismos também foram construídas no trabalho de Zhang *et al.* [63]. A molécula 16S rRNA é um componente da subunidade 30S de ribossomos de procariotos, com aproximadamente 1500 nucleotídeos, tem papel estrutural, agindo como determinante das posições da proteína ribossomal. Essa molécula é altamente conservada entre espécies de bactérias e arqueias e, por isso mesmo, muito usada na construção de árvores filogenéticas. A título de comparação com as árvores filogenéticas baseadas em vias metabólicas, árvores filogenéticas baseadas na molécula 16S rRNA dos organismos foram construídas. Inicialmente, essas moléculas foram obtidas do *Ribosomal Database Project-II* [18] e um alinhamento das 171 16S rRNA sequências genéticas foi obtido com o programa CLUSTALW [59]. Em seguida, o programa DNADIST do pacote PHYLIP [27] foi usado para construir a matriz de distâncias baseada nesse alinhamento. Por fim, foram selecionados os organismos correspondentes aos das árvores filogenéticas baseadas em vias metabólicas, juntamente com as matrizes de distâncias correspondentes e árvores filogenéticas foram construídas usando o programa NEIGHBOR do pacote PHILIP [27].

As similaridades entre as árvores filogenéticas construídas, baseadas nas vias metabólicas dos organismos e também na molécula 16S rRNA, foram computadas usando o método de Penny e Hendy [52] e mostraram que são muito próximas.

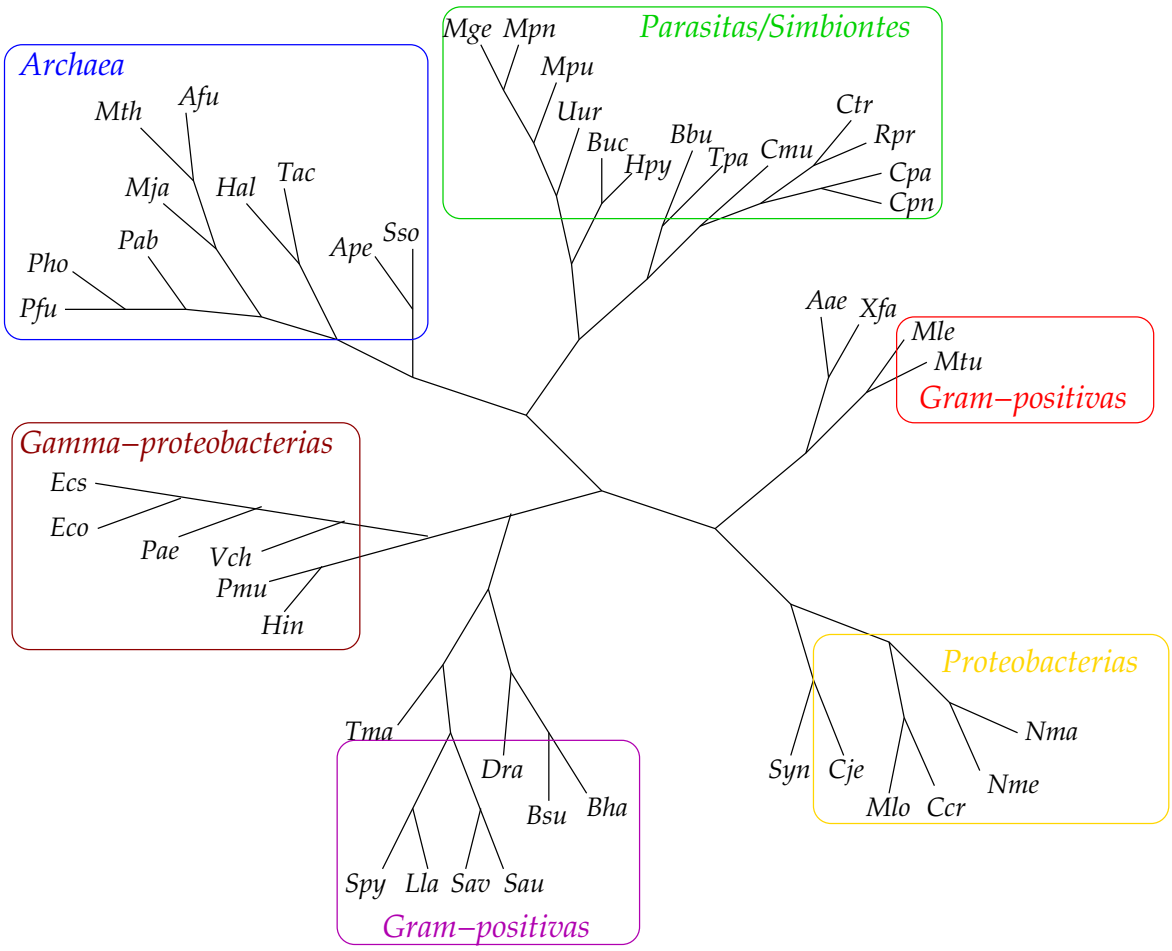

Figura 4.4 A árvore filogenética baseada nas vias metabólicas dos 47 organismos do experimento realizado por Zhang *et al.* [63].

Capítulo 5

Considerações finais

Neste trabalho, estudamos dois métodos para construção de árvores filogenéticas baseados em vias ou redes metabólicas. No primeiro deles, apresentamos um método baseado em distâncias de Heymans & Singh [35], onde uma via metabólica ou um conjunto de vias metabólicas é fixada(o) em um conjunto de organismos, modeladas como grafos de enzimas dos organismos e distâncias entre pares desses grafos são obtidas a partir da análise da similaridade entre seus vértices e das relações estruturais entre eles. Uma matriz de distâncias entre esses grafos é assim obtida e uma árvore filogenética é finalmente construída, usando um método de construção de árvores filogenéticas baseado em distâncias. Uma implementação desse método foi obtida e experimentos foram realizados sobre um conjunto pequeno de organismos, produzindo uma árvore filogenética distinta daquela produzida por um método baseado em distâncias que toma como entrada sequências bem conservadas como as sequências de 16S rRNA dos mesmos organismos. No outro trabalho estudado de Zhang *et al.* [63] apresentamos um método que seleciona um conjunto de vias metabólicas e, para cada uma delas, verifica inicialmente a presença de tal via metabólica nos organismos de interesse. Do mesmo modo, as vias metabólicas são modeladas como grafos de enzimas. A partir desse conjunto de organismos e dessas vias metabólicas, o método constroi uma árvore filogenética a partir das distâncias computadas entre os pares de grafos de enzima, considerando também a similaridade entre seus vértices e as relações estruturais entre eles. Por fim, obtém-se uma árvore filogenética completa usando as árvores filogenéticas individuais das vias metabólicas nos organismos usando o método dos quartetos.

Dezenas de métodos para construção de árvores filogenéticas baseados em vias ou redes metabólicas de organismos têm sido desenvolvidos. Dentre os mais antigos, podemos classificá-los como aqueles em que uma via ou rede metabólica é fixada, alguma análise biológica dessa via é realizada nos organismos de interesse – tipicamente uma análise que determina algum tipo de similaridade entre a via fixa e as outras vias de interesse – e então uma árvore filogenética é construída. Assim como o trabalho de Heymans & Singh [35], que apresentamos no capítulo 3, outros trabalhos mais recentes também se enquadram nesta classificação [22, 30, 31]. Outros métodos procuram capturar as modificações gradativas nos sistemas biológicos, considerando que o mecanismo evolutivo pode causar alterações funcionais que forçam um sistema a se adaptar a novas configurações, o que pode se refletir, por exemplo, na conformação espacial das vias ou redes metabólicas. Uma abordagem como esta é apresentada no trabalho de Zhang *et al.* [63], que apresentamos no capítulo 4, dentre outros existentes na literatura e também mais recentes [1, 32, 45, 53, 51, 13].

Pensando em trabalhos futuros, uma possivel proposta seria agrupar mais características das redes metabólicas como fatores a serem analisados na construção das árvores, além de continuar analisando a estrutura das redes de outras perspectivas. Em [17], uma análise estrutural de uma rede metabólica é realizada pelo estudo da hierarquia, das informações contidas e da ortologia das enzimas gerando uma árvore filogenética. Em [49], as estruturas são analisadas a partir de uma abordagem *kernel-based* usando uma matriz de rotulação e outra de adjacência para assim construir uma árvore filogenética. Em [16], uma abordagem diferente é apresentada pelo uso da máxima parcimônia para inferência de estados ancestrais em uma árvore filogenética das vias metabólicas nos organismos.

Referências Bibliográficas

[1] T. Arodź. Clustering organisms using metabolic networks. *ICCS - International Conference on Computational Science – Part II*, 2008.

[2] S. Atran. Cognitive foundations of natural history: towards an anthropology of science. *Cambridge Univ. Press*, 1990.

[3] F. Ay and T. Kahveci. Functional similarities of reaction sets in metabolic pathways. *Proceedings of the First ACM International Conference on Bioinformatics and Computational Biology*, 2010.

[4] H.-J. Bandelt and A. Dress. Reconstructing the shape of a tree from observed dissimilarity data. *dvances in Applied Mathematics*, 7:309–343, 1986.

[5] A. Banerjee. Structural distance and evolutionary relationship of networks. *BioSystems*, 2012.

[6] A. Ben-Dor, B. Chor, D. Graur, R. Ophir, and D. Pelleg. Constructing phylogenies from quartets: elucidation of Eutherian superordinal relationships. *Journal of Computational Biology*, 5(3):377–390, 1998. Extended Abstract.

[7] A. Ben-Dor, B. Chor, D. Graur, R. Ophir, and D. Pelleg. From four-taxon trees to phylogenies: the case of mammalian evolution. In *Proceedings of the 2nd Annual International Conference on Computational Molecular Biology (RECOMB 98)*, pages 9–19, 1998.

[8] V. Berry, D. Bryant, T. Jiang, P. E. Kearney, M. Li, T. Wareham, and H. Zhang. A practical algorithm for recovering the best supported edges of an evolutionary tree. In *Proceedings of the 11th ACM-SIAM Symposium on Discrete Algorithms (SODA 2000)*, pages 287–296. ACM Press, 2000.

[9] V. Berry and O. Gascuel. Inferring evolutionary trees with strong combinatorial evidence. *Theoretical Computer Science*, 240(240):271–298, 2000.

[10] V. Berry, T. Jiang, P. E. Kearney, M. Li, and T. Wareham. Quartet cleaning: improved algorithms and simulations. In *Proceedings of the 7th European Symposium on Algorithms (ESA 99)*, volume 1643 of *Lecture Notes in Computer Science*, pages 313–324. Springer-Verlag, 1999.

[11] P. Buneman. The recovery of trees from measures of dissimilarity. In F. R. Hodson, D. G. Kendall, and P. Tautu, editors, *Anglo-Romanian Conference on Mathematics in the Archaeological and Historical Sciences*, pages 387–395, Edinburgh, 1971. Edinburgh University Press.

[12] C.-W. Chang, P.-C. Lyu, and M. Arita. Reconstructing phylogeny from metabolic substrate-product relationships. *BMC Bioinformatics*, 12:27–32, 2011.

[13] X. Chang, Z. Wang, P. Hao, Y.-Y. Li, and Y.-X. Li. Exploring mitochondrial evolution and metabolism organization principles by comparative analysis of metabolic networks. *Genomics*, 95(95):339–344, 2010.

[14] G. Chartrand and O. Oellermann. *Applied and Algorithmic Graph Theory.* International Series in Pure and Applied Mathematics. McGraw-Hill, Inc, Hightstown, NJ 08520, 1993.

[15] B. Chor. From quartets to phylogenetic trees. In B. Rovan, editor, *Proceedings of the 25th Conference on Current Trends in Theory and Practice of Informatics (SOFSEM 98)*, volume 1521 of *Lecture Notes in Computer Science*, pages 36–53, Jasná, Slovakia, November 1998. Springer-Verlag.

[16] J. C. Clemente, K. Ikeo, G. Valiente, and T. Gojobori. Optimized ancestral state reconstruction using sankoff parsimony. *BMC Bioinformatics*, 10(51), 2009.

[17] J. C. Clemente, K. Satou, and G. Valiente. Phylogenetic reconstruction from non-genomic data. *Bioinformatics*, 23:e110–e115, 2006.

[18] J. R. Cole, B. Chai, R. J. Farris, Q. Wang, S. A. Kulam, D. M. McGarrell, G. M. Garrity, and J. M. Tiedje. The ribosomal database project (rdp-ii): sequences and tools for high-troughput rrna analys. *Nucleic Acids Res*, 2005.

[19] N. Colonius and H. H. Schulze. Tree structures for proximity data. *British Journal of Mathematical Statistics and Psychology*, 34:425–453, 1981.

[20] M. Csürös and M.-Y. Kao. Recovering evolutionary trees through harmonic greedy triplets. In *Proceedings of th 10th ACM-SIAM Symposium on Discrete Algorithms (SODA 99)*, pages 261–270. ACM Press, 1999.

[21] M. Csürös and M.-Y. Kao. Provably fast and accurate recovery of evolutionary trees through harmonic greedy triplets. *SIAM Journal on Computing*, 31(1):306–322, 2001.

[22] C. Cunchillos and G. Lecointre. Evolution of amino acid metabolism inferred through cladistic analysis. *Journal of Biological Chemistry*, 278(48):47960–47970, 2003.

[23] G. Ding, Z. Yu, J. Zhao, Z. Wang, Y. Li, X. Xing, C. Wang, L. Liu, and Y. Li. Tree of life based on genome context networks. *PLoS ONE*, 3(10), 2008.

[24] P. L. Erdös, M. A. Steel, L. A. Székely, and T. J. Warnow. Constructing big trees from short sequences. In *Proccedings of the 24th International Conference on Automata, Languages, and Programming (ICALP 97)*, volume 1256 of *Lecture Notes in Computer Science*, pages 827–837. Springer-Verlag, 1997.

[25] P. L. Erdös, M. A. Steel, L. A. Székely, and T. J. Warnow. A few logs suffice to build (almost) all trees: Part I. *Random Structures and Algorithms*, 14(2):153–184, 1999.

[26] P. L. Erdös, M. A. Steel, L. A. Székely, and T. J. Warnow. A few logs suffice to build (almost) all trees: Part II. *Theoretical Computer Science*, 221:77–118, 1999.

[27] J. Felsenstein. Phylogeny programs. `http://bit.ly/kwcEa`, 2010. (último acesso em agosto de 2010).

[28] P. Feofiloff, Y. Kohayakawa, and Y. Wakabayashi. *Uma Introdução Sucinta à Teoria dos Grafos*. II Bienal da SBM (minicurso). Sociedade Brasileira de Matemática, Salvador, BA, Brasil, 2009.

[29] W. M. Fitch. A non-sequential method for constructing trees and hierarchical classifications. *Journal of Molecular Evolution*, 18:30–37, 1981.

[30] C. Forst, C. Flamm, I. Hofacker, and P. Stadler. Algebraic comparison of metabolic networks, phylogenetic inference, and metabolic innovation. *BMC Bioinformatics*, 7(1):67, 2006.

[31] C. Forst and K. Schulten. Phylogenetic analysis of metabolic pathways. *Journal of Molecular Evolution*, 52:471–489, 2001.

[32] S. Freilich, L. Goldovsky, C. A. Ouzounis, and J. M. Thornton. Metabolic innovations towards the human lineage. *BMC Evolutionary Biology*, 8(247), 2008.

[33] J. Gramm and R. Niedermeier. Minimum quartet inconsistency is fixed parameter tractable. In A. Amir and G. M. Landau, editors, *Proceedings of the 12th Annual Symposium on Combinatorial Pattern Matching (CPM 2001)*, volume 2089 of *Lecture Notes in Computer Science*, pages 241–256, Jerusalem, Israel, July 2001. Springer-Verlag.

[34] M. Heymans and A. Singh. Deriving phylogenetic trees from the similarity analysis of metabolic pathways. Technical Report TR 2002-33, University of California at Santa Barbara, Santa Barbara, CA 93106, December 2002.

[35] M. Heymans and A. Singh. Deriving phylogenetic trees from the similarity analysis of metabolic pathways. *Bioinformatics*, 19(1):i138–i146, 2003.

[36] Y. Huang, K. Jiang, and J. Robertson. A method of biological pathway similarity search using high performance computing. *Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine & Biology Society*, 2009.

[37] T. Jiang, P. E. Kearney, and M. Li. Some open problems in computational molecular biology. *Journal of Algorithms*, 34:194–201, 2000.

[38] M. Kanehisa and S. Goto. Kegg: kyoto encyclopedia of genes and genomes. *Nucleic Acids Res*, 28(I)(28(I)):27–30, 2000.

[39] P. E. Kearney. The ordinal quartet method. In *Proceedings of the 2nd Annual International Conference on Computational Molecular Biology (RECOMB 98)*, pages 125–134, 1998.

[40] J. O. Korbel, B. Snel, M. A. Huynen, and P. Bork. Shot: a server for construction of genome phylogenies. *Trends Genetics*, 2002.

[41] H. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[42] V. Lacroix, L. Cottret, P. Thébault, and M. Sagot. An introduction to metabolic networks and their structural analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 5(4):594–617, 2008.

[43] J. Larson. Reason and experience. the representation of natural order in the work of carl von linne. *Univ. of California Press*, 11:265–267, 1973.

[44] D. Maddison, K. Schulz, and W. Maddison. The tree of life project. *Zootaxa*, 1668:19–40, 2007.

[45] A. Mano, T. Tuller, O. Béjà, and R. Y. Pinter. Comparative classification of species and the study of pathway evolution based on the alignment of metabolic pathways. *BMC Bioinformatics*, 11(Suppl 1):S38, 2010.

[46] A. Mazurie, D. Bonchev, B. Schwikowski, and G. A. Buck. Phylogenetic distances are encoded in networks of interacting pathways. *Bioinformatics*, 24(22):2579–2585, 2008.

[47] A. Mithani, G. M. Preston, and J. Hein. A bayesian approach to the evolution of metabolic networks on a phylogeny. *PLoS Computational Biology*, 6(8):1–19, 2010.

[48] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38, 1957.

[49] S. J. Oh, J. Joung, J.-H. Chang, and B. Zhang. Construction of phylogenetic trees by kernel-based comparative analysis of metabolic networks. *BMC Bioinformatics*, 7(284), 2010.

[50] R. D. M. Page and E. C. Holmes. Molecular evolution: A phylogenetic approach. *Blackwell Science*, 3(3):178–190, 1998.

[51] M. Parter, N. Kashtan, and U. Alon. Environmental variability and modularity of bacterial metabolic networks. *BMC Evolutionary Biology*, 7(169), 2007.

[52] D. Penny and M. D. Hendy. The use of tree comparation metrics. *Systematic Zoology*, 34:75–90, 1985.

[53] J. M. Peregrín-Alvarez, C. Sanford, and J. Parkinson. The conservation and evolutionary modularity of metabolism. *Genome Biology*, 10(6), 2009.

[54] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, 1987.

[55] S. Sattath and A. Tversky. Additive similarity trees. *Psychometrika*, 42:319–345, 1977.

[56] K. St. John, T. J. Warnow, B. M. E. Moret, and L. Vawter. Performance study of phylogenetic methods: (unweighted) quartet methods and neighbor-joining. In *Proceedings of the 12th ACM-SIAM Symposium on Discrete Algorithms (SODA 2001)*, pages 196–206. ACM Press, 2001.

[57] M. A. Steel. The complexity of reconstructing trees from qualitative characters and subtrees. *Journal of Classification*, 9:91–116, 1992.

[58] K. S. Strimmer and A. von Haeseler. Quartet puzzling: a quartet maximum-likelihood method for reconstructing tree topologies. *Molecular Biology and Evolution*, 13(7):964–969, 1996.

[59] J. Thompson, D. Higgins, and T. Gibson. Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res*, 22:4673–4680, 1994.

[60] Z. Wang, Q. Chen, and L. Liu. Relationship between topology and functions in metabolic network evolution. *Chinese Science Bulletin*, 54(5):776–782, 2009.

[61] S. J. Willson. Measuring inconsistency in phylogenetic trees. *Journal of Theoretical Biology*, 190:15–36, 1998.

[62] L. A. Zager and G. C. Verghese. Graph similarity scoring and matching. *Applied Mathematics Letters*, 21(21):86–94, 2008.

[63] Y. Zhang, S. Li, G. Skogerbø, Z. Zhang, X. Zhu, Z. Zhang, S. Sun, H. Lu, B. Shi, and R. Chen. Phylogenetic properties of metabolic pathway topologies as revealed by global analysis. *BMC Bioinformatics*, 7(1):252, 2006.

[64] J.-B. Zhao and G. Lin. An algorithm for extracting subgraph of specific species from metabolic pathway. *IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications*, pages 74–79, 2010.

[65] T.-T. Zhou, K.-F. Yung, C.-C. K. Chan, Z.-H. Wang, Y.-P. Zhu, and F.-C. He. Metagen: a promising tool for modeling metabolic networks from kegg. *Progress in Biochemistry and Biophysics*, 37(1):63–68, 2010.