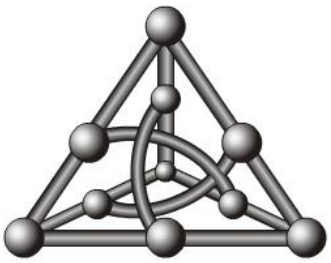
Medianas em genômica comparativa

Helmuth Ossinaga Martines da Silva

Dissertação de mestrado

Orientador: Fábio Henrique Viduani Martinez

Área de concentração: Teoria da Computação



Faculdade de Computação
Universidade Federal de Mato Grosso do Sul

Abril de 2022

Medianas em genômica comparativa

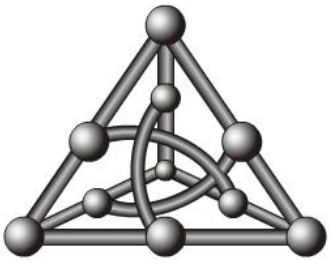
Helmuth Ossinaga Martines da Silva

Dissertação de mestrado

Orientador: Fábio Henrique Viduani Martinez

Área de concentração: Teoria da Computação

Dissertação apresentada à Faculdade de Computação da Universidade Federal de Mato Grosso do Sul, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para a obtenção do título de Mestre.



Faculdade de Computação
Universidade Federal de Mato Grosso do Sul

Abril de 2022

Agradecimentos

Agradeço primeiramente à Deus por me conceder a oportunidade de realizar uma pós-graduação e por até aqui ter me sustentado em meio a uma pandemia, foi uma experiência ímpar em minha vida.

À minha família e amigos, em especial minha mãe Sandra e meu pai Helmut, por estarem sempre me apoiando e incentivando em tudo quanto faço.

Ao meu orientador Fábio, por ter me ensinado e orientado com bom ânimo, paciência e maestria durante todo esse tempo de mestrado.

E à todos aqueles que me apoiaram com palavras e ações, também foram determinantes na minha vida nesse período.

Resumo

A inferência de genomas ancestrais é uma tarefa clássica em genômica comparativa. Aqui, estudamos o problema da mediana de genomas tal que, dado um conjunto de três ou mais genomas, queremos encontrar um novo genoma que minimize a soma das distâncias par a par entre esse e os genomas dados. A distância representa a quantidade de evolução observada no nível do genoma, para a qual determinamos o número mínimo de operações de rearranjos necessárias para transformar um genoma em outro. Para quase todas as operações de rearranjo conhecidas, o problema da mediana é NP-difícil, com exceção da operação *single-cut-or-join* (SCJ) que pode ser resolvido eficientemente para genomas multicromossomais circulares e mistos. Neste projeto, estudamos o problema da mediana sob uma medida de rearranjo restrita chamada distância-c_4, que é estreitamente relacionada à distância SCJ e à DCJ *(double-cut-and-join)*. Identificamos limitantes precisos e *decomposers* da mediana-c_4 e implementamos algoritmos para a sua construção, dois algoritmo exatos baseados em PLI (Programação Linear Inteira) e três heurísticas combinatórias. Posteriormente, realizamos experimentos com conjunto de dados simulados. Nossos resultados sugerem que a distância c_4 é útil para estudo do problema da mediana de genomas, de perspectiva teórica e prática.

Palavras-chave: problema da mediana, otimização, programação linear inteira, heurísticas

Abstract

Ancestral genome inference is a classic task in comparative genomics. Here, we study the genome median problem, a related computational problem which, given a set of three or more genomes, asks to find a new genome that minimizes sum of pairwise distances between it and the given genomes. The distance stands for the amount of evolution observed at the genome level, for which we determine the minimum number of rearrangement operations necessary to transform one genome into the other. For almost all rearrangement operations the median problem is NP-hard, with the exception of the SCJ median that can be constructed efficiently for multichromosomal circular and mixed genomes. In this work we study the median problem under a restricted rearrangement measure called c_4-distance, which is closely related to the breakpoint and the DCJ distance. We identify tight bounds and decomposers of the c_4-median and develop algorithms for its construction, two exacts ILP-based and three combinatorial heuristics. Subsequently, we perform experiments on simulated data sets. Our results suggest that the c_4-distance is useful for the study the genome median problem, from theoretical and practical perspectives.

Keywords: median problem, optimization, integer linear programming, heuristics

Sumário

Capítulo 1

Introdução

A identificação de relações entre organismos vivos, buscando similaridades entre si, visando descobertas para cura de doenças e obtenção de novas informações a respeito dos seres vivos, faz parte da genômica comparativa [8]. Neste projeto, estudamos o problema de encontrar uma mediana de genomas que evoluem por mutações em larga escala, conhecidas também por rearranjos. Essas mutações alteram a ordem e a orientação dos marcadores dentro e entre sequências cromossômicas. É comum supor que o cenário evolutivo subjacente é parcimonioso, dessa forma o número mínimo de rearranjos entre dois genomas fornece uma noção de distância de rearranjo. Embora as distâncias de rearranjo possam ser computadas eficientemente em alguns cenários, encontrar uma mediana de três ou mais genomas, ou seja, um genoma que minimize a soma das distâncias de rearranjo entre si e os genomas dados, é computacionalmente intratável para maioria das distâncias de rearranjo simples.

A distância entre dois genomas depende da operação de rearranjo escolhida. Por exemplo, o número de *breakpoints* entre dois genomas, ou seja, o número de pares de marcadores que aparecem consecutivamente em um genoma mas não no outro, dá origem a uma distância de rearranjo simples. Enquanto que, a rigor, a distância de *breakpoint* não é de fato uma operação de rearranjo [6], outras distâncias são, tais como a distância de *double-cut-and-join* (DCJ) [16]. Uma operação DCJ quebra um genoma, representado por um conjunto de sequências de marcadores, em duas posições arbitrárias e posteriormente reconecta as quatro pontas abertas assim criadas em uma nova combinação.

Quase todas as distâncias de rearranjo conhecidas podem ser computadas eficientemente em tempo linear sob a suposição de que os marcadores são únicos em cada genoma [6, 16, 2]. Entretanto, considerando um passo à frente, a construção de uma mediana de três genomas é NP-difícil sob quase todas as distâncias de rearranjo, incluindo DCJ [13], com duas notáveis exceções: a distância de *breakpoint* e a distância *single-cut-or-join* (SCJ) são tratáveis para genomas multicromossomais circulares e mistos [13, 5].

O grafo de *breakpoint* é uma estrutura de dados muito usada no cálculo de distâncias de rearranjo e, ao construí-lo, o número de ciclos neste grafo desempenha um papel essencial. Por exemplo, os ciclos de comprimento 2 representam adjacências no grafo de *breakpoint*, que são a contrapartida dos *breakpoints*. Ciclos maiores representam a exigência de uma ou mais operações DCJ para transformar um genoma em outro. Assim, para calcular a distância de *breakpoint*, os ciclos de comprimento 2 são contados (e esta quantidade é então subtraída do número de marcadores dos dois genomas). Da mesma forma, para calcular a distância DCJ, o número de

ciclos de qualquer comprimento é contado [13].

Neste projeto, estudamos a distância-c_4 entre dois genomas, que se baseia no número de ciclos de comprimento até 4 no grafo de *breakpoint*. Em outras palavras, somente são contados aqueles ciclos que requerem no máximo uma operação DCJ para serem transformados em adjacências. Aqui, abordamos o problema da mediana-c_4, ou seja, a construção de um novo genoma que minimiza a soma das distâncias-c_4 par a par entre estes e cada membro de um conjunto de três ou mais genomas dados. Em particular, estamos interessados em encontrar a mediana-c_4 para três genomas.

É importante destacar que a distância-c_4 não é uma métrica, uma vez que a desigualdade triangular não se mantém nesta medida, ou seja, tendo um conjunto de três genomas $\mathbf{\Pi} = \{\Pi_1, \Pi_2, \Pi_3\}$ e a mediana-c_4 Γ, a distância-c_4 de (Γ, Π_i) pode ser maior que a soma entre as distâncias-c_4 de (Γ, Π_j). e (Γ, Π_k), sendo $i \neq j \neq k$. Isso geralmente acontece quando há muitas adjacências em comum entre um par de genomas em $\mathbf{\Pi}$.

Este documento está estruturado da seguinte forma. O capítulo 2 fornece as definições básicas e notações, no capítulo 3 é apresentado o problema da mediana, a formulação em teoria dos grafos, os limitantes e os *decomposers*, que são blocos de construção da mediana. No capítulo 4 descrevemos os algoritmos para calcular a mediana-c_4, incluindo dois algoritmos exatos baseados em PLI (Programação Linear Inteira) e três heurísticas. No capítulo 5 são apresentados os resultados experimentais para instâncias simuladas e as considerações finais no capítulo 6.

Capítulo 2

Preliminares

2.1 Definições básicas

Um **marcador** é um fragmento orientado de DNA. Um marcador g possui duas extremidades distintas chamadas **cauda** e **cabeça**, representadas, respectivamente, por g^t e g^h. Se a orientação do marcador g for $g^h g^t$, a sua representação será $\overline{g}$, caso contrário será g. O **cromossomo** é uma sequência de marcadores e pode ser linear ou circular. O cromossomo linear tem duas extremidades e cada uma delas é um **telômero**. A **adjacência** em um cromossomo é composta por duas extremidades de marcadores. O cromossomo circular é denotado por uma sequência de marcadores e delimitado por parênteses. A Figura 2.1 mostra a representação de um cromossomo linear e um circular.

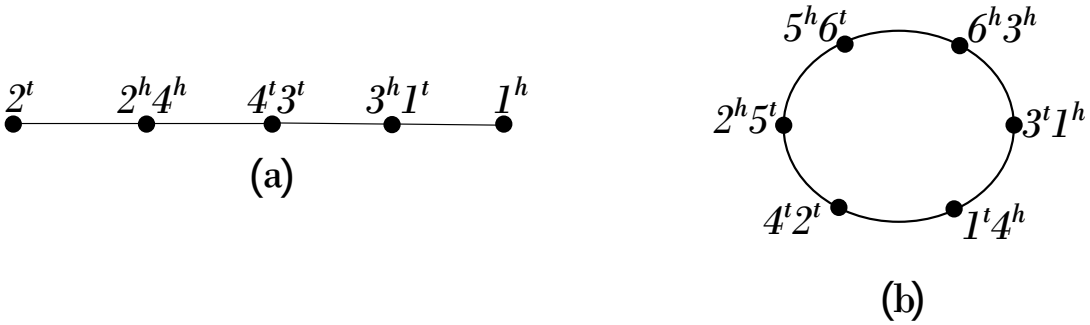


Figura 2.1: (a) Um cromossomo linear $2\ \overline{4}\ 3\ 1$ com as adjacências $2^h4^h, 4^t3^t$ e 3^h1^t, e os telômeros 2^t e 1^h. (b) Um cromossomo circular $(2\ 5\ 6\ \overline{3}\ \overline{1}\ \overline{4})$ com as adjacências $2^h5^t, 5^h6^t, 6^h3^h, 3^t1^h, 1^t4^h$ e 4^t2^t.

Um **genoma** é uma coleção de cromossomos. O genoma é **unicromossomal** quando contém apenas um cromossomo e **multicromossomal** quando contém mais de um cromossomo, como está representado na Figura 2.2.

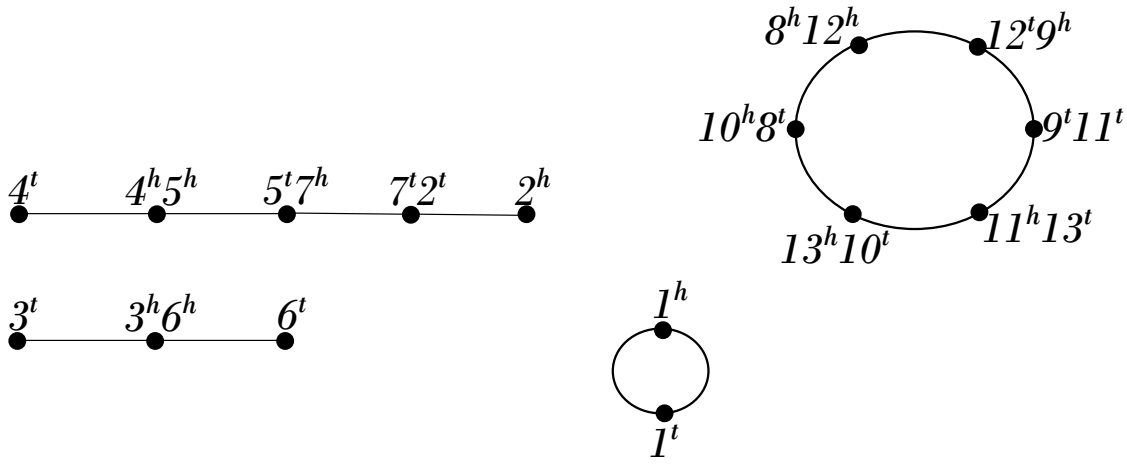


Figura 2.2: Genoma $\{(1), 4\ \overline{5}\ \overline{7}\ 2, (10\ 8\ \overline{12}\ \overline{9}\ 11\ 13), 3\ \overline{6}\}$ contendo dois cromossomos lineares e dois cromossomos circulares.

2.2 Distâncias clássicas

Pode-se calcular uma distância de rearranjo entre dois genomas com suporte de uma estrutura de dados conhecida como **grafo de *breakpoint*** [1]. Seja $\mathcal{M}$ um conjunto de n marcadores e $\mathcal{M}_x$ o conjunto das extremidades de todos os marcadores em $\mathcal{M}$, com $|\mathcal{M}_x| = 2n$. Para dois genomas A e B, cada um com n marcadores de $\mathcal{M}$, o grafo de *breakpoint* $BG(A, B)$ de $A \cup B$ tem o conjunto de vértices representado pelas extremidades de $\mathcal{M}_x$ e as arestas são as adjacências dos genomas A e B, com arestas-A e arestas-B, respectivamente. A Figura 2.3 mostra o grafo de *breakpoint* dos genomas A e B.

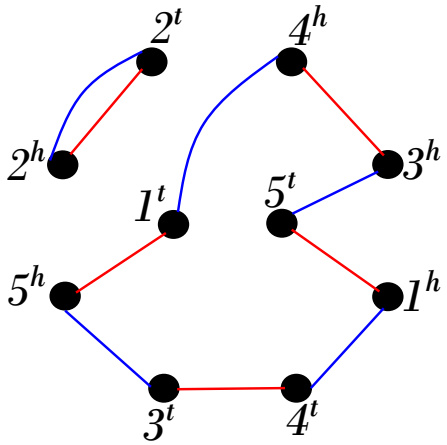


Figura 2.3: Genomas $A = \{(2), (1\ 5), (4\ \overline{3})\}$ e $B = \{(3\ 5), (2), (1\ 4)\}$ tendo, cada um, $n = 5$ marcadores. As arestas-A de cor vermelho representam as adjacências 2^h2^t, 3^h4^h, 3^t4^t, 1^h5^t e 1^t5^h. As arestas-B de cor azul representam as adjacências 1^t4^h, 3^h5^t, 1^h4^t, 3^t5^h e 2^h2^t.

O grafo de *breakpoint* pode conter vértices de grau 0, 1 ou 2 resultando em um conjunto de ciclos e caminhos. Dados dois genomas A e B, com o mesmo conjunto de n marcadores, a **distância de *breakpoint*** [13] é definida da seguinte forma:

$$\mathrm{d}_{\mathrm{BKP}}(A, B) = n - a - t/2$$

onde a é o número de adjacências em comum entre A e B e t é o número de telômeros em comum entre A e B. Como exemplo, na Figura 2.3, os genomas são circulares e com isso, $a = 1$ e $t = 0$.

Segundo [5], a distância de *breakpoint* é equivalente à distância *single-cut-or-join* (SCJ) em genomas circulares, pois ambas utilizam as adjacências em comum entre dois genomas em suas fórmulas. Como a é o número de adjacências em comum em $BG(A, B)$, cada uma dessas adjacências representa um ciclo de comprimento 2 em $BG(A, B)$ e pode ser denotado por $c_2 = a$ (na Figura 2.3 a adjacência $2^t 2^h$ é comum nos genomas A e B formando um ciclo de comprimento 2). Chamamos um ciclo de comprimento j de **j-ciclo**.

Por outro lado, se c é o número de ciclos (de qualquer comprimento) e e é o número de caminhos de comprimento par de arestas em $BG(A, B)$, a distância *double-cut-and-join* (DCJ) [16] entre A e B é:

$$\mathrm{d}_{\mathrm{DCJ}}(A, B) = n - c - e/2.$$

As distâncias de *breakpoint*/SCJ e DCJ podem ser computadas eficientemente [2, 13, 5].

2.3 Genomas multicromossomais circulares

Cromossomos e plasmídeos de organismos unicelulares como bactérias e arqueas, DNA mitocondrial dentro de células eucarióticas e DNA cloroplástico em plantas são exemplos de cromossomos/genomas circulares e motivam o estudo de mediana de genomas circulares. Além disso, por tratarmos de maximização de ciclos e o problema da mediana é NP-difícil para genomas multicromossomais, desejamos simplificar a distância utilizando somente adjacências. Portanto, serão considerados os genomas multicromossomais apenas com cromossomos circulares.

Note que se dois genomas A e B têm apenas cromossomos circulares, temos $\mathrm{d}_{\mathrm{BKP}}(A, B) = n - c_2$ e $\mathrm{d}_{\mathrm{DCJ}}(A, B) = n - c = n - c_2 - c_4 - c_6 \ldots$, onde c_j denota o número de j-ciclos no grafo de *breakpoint* de A e B.

Capítulo 3

Problema da mediana

3.1 Definição do problema

Seja $\mathcal{M}$ o conjunto de n marcadores e $\mathbf{\Pi}$ um conjunto de $p \geq 3$ genomas sobre $\mathcal{M}$. O problema da mediana em $\mathbf{\Pi}$ requer encontrar um genoma Γ com n marcadores de $\mathcal{M}$ em que a soma das distâncias entre Γ e cada genoma em $\mathbf{\Pi}$, sob uma operação de rearranjo, é mínima. Se a operação for *breakpoint*/SCJ, então a mediana pode ser computada em tempo polinomial no tamanho dos genomas de entrada [5]. Entretanto, para a operação DCJ, o problema da mediana é NP-difícil, mesmo para $p = 3$ [13, 3].

A **distância-c_4**, denotada por d_4, entre Π_i e Π_j é dada por $d_4(\Pi_i, \Pi_j) = n - c_2 - c_4$, onde n é o número de marcadores em $\mathcal{M}$ e em Π_i e Π_j, e c_ℓ é o número de ℓ-ciclos em $BG(\Pi_i, \Pi_j), \ell \in \{2, 4\}$.

Seja $\mathbf{\Pi} = \{\Pi_1, \ldots, \Pi_p\}$ um conjunto de $p \geq 3$ genomas e Γ um genoma. O **custo-c_4** $K(\mathbf{\Pi}, \Gamma)$ de Γ dado $\mathbf{\Pi}$ é

$$K(\mathbf{\Pi}, \Gamma) = \sum_{\Pi_i \in \mathbf{\Pi}} d_4(\Pi_i, \Gamma).$$

Dizemos que o genoma Γ é a **mediana-c_4** do conjunto de $p \geq 3$ genomas $\mathbf{\Pi}$ se Γ minimiza $K(\mathbf{\Pi}, \Gamma)$. Para um determinado conjunto de genomas $\mathbf{\Pi}$, denota-se por $K^\star(\mathbf{\Pi})$ o valor da mediana-c_4:

$$K^\star(\mathbf{\Pi}) = \min_{\Gamma}\{K(\mathbf{\Pi}, \Gamma)\}.$$

Assim, podemos estabelecer formalmente o seguinte:

Problema MEDIANA-$c_4(\mathbf{\Pi})$: Dado $p \geq 3$ genomas em $\mathbf{\Pi}$, cada um com n marcadores de $\mathcal{M}$, encontrar um genoma Γ com n marcadores de $\mathcal{M}$ tal que $K^\star(\mathbf{\Pi}) = K(\mathbf{\Pi}, \Gamma)$.

Particularmente estamos interessados na versão mais simples do problema MEDIANA-c_4, onde $p = 3$.

3.2 Formulação em teoria dos grafos

Podemos reformular o problema MEDIANA-c_4 em termos de grafos. Atribuímos cada extremidade de um marcador do conjunto de n marcadores $\mathcal{M}$ a um vértice em um grafo G e assim $|V(G)| = 2n$. Uma adjacência em um determinado genoma Π_i é representada como uma aresta em G com cor i, ou seja, uma adjacência uv em Π_i é uma aresta uv em G com cor i. Além disso, um genoma Π_i é um grafo 1-regular sobre $V(G)$, isto é, um emparelhamento perfeito em G. Dessa forma, G é um grafo p-regular e p-aresta-colorido. Note que G é uma generalização do grafo de *breakpoint* [1] para pelo menos 3 genomas, e o chamamos de **grafo de *breakpoint* estendido** para $\mathbf{\Pi}$, denotado por $BG_x(\mathbf{\Pi})$. Logo, G é um grafo p-regular e p-aresta-colorido.

Seja Γ um subconjunto de pares não ordenados de $V(G)$, tal que $|\Gamma| = n$ e $e \cap f = \emptyset$ para cada par e, f em Γ. Dessa forma, Γ é um grafo 1-regular sobre $V(G)$. Veja a Figura 3.1 como exemplo.

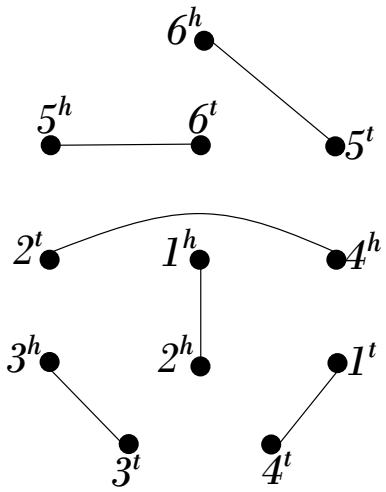


Figura 3.1: Grafo Γ é representado pelos pares de vértices $\{3^t3^h,\ 1^h2^h,\ 1^t4^t,\ 2^t4^h,\ 5^h6^t$ e $5^t6^h\}$ resultando em um grafo 1-regular.

Definimos $G^\Gamma := G + \Gamma$ como um multigrafo tal que $V(G^\Gamma) = V(G)$ e $E(G^\Gamma) = E(G) \cup \Gamma$. Portanto, G^Γ é um multigrafo $(p+1)$-aresta-colorido. Dizemos que um ciclo em G^Γ é **i-colorido** se suas arestas tiverem cores alternando entre i e $p+1$, $i \in \{1, 2, \ldots, p\}$. Dizemos também que um ciclo em G é **bicolorido** se as arestas tiverem cores alternando entre i e j, $i \neq j$, $i, j \in \{1, \ldots, p\}$. Veja a Figura 3.2 que mostra um exemplo de G^Γ, ciclo bicolorido e ciclos i-colorido.

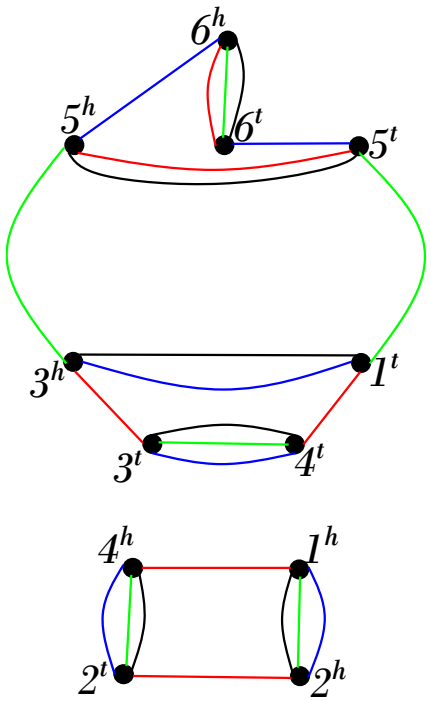


Figura 3.2: Grafo $G^\Gamma := G+\Gamma$, com $\Gamma = \{(1\ \overline{2}\ \overline{4}\ 3), (5), (6)\}$ e os emparelhamentos perfeitos $\Pi_1 = \{(1\ \overline{4}), (2), (3), (5), (6)\}$, $\Pi_2 = \{(1\ \overline{2}\ \overline{4}\ 3), (5\ \overline{6})\}$ e $\Pi_3 = \{(1\ \overline{2}\ \overline{4}\ 3\ \overline{5}), (6)\}$. Há um ciclo bicolorido com as arestas 1^h4^h, 4^h2^t, 2^t2^h e 2^h1^h alternando entre cores vermelho e azul. Também temos um exemplo de ciclo i-colorido de comprimento 2 com aresta 1^t3^h e de comprimento 4 com as arestas $\{3^h5^h, 5^h5^t, 1^t5^t, 1^t3^h\}$ em que há alternância das cores i e $p+1$.

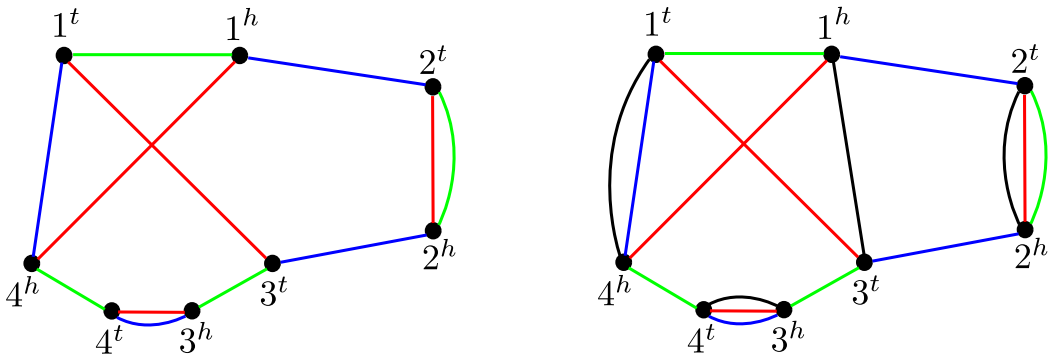
Denotamos por $k(G^\Gamma)$ o número de 2- e 4-ciclos i-coloridos em G^Γ. E denotamos por $k(G)$ o número máximo de 2- e 4-ciclos i-coloridos em G^Γ para todos os possíveis grafos 1-regulares com conjunto de vértices $V(G)$:

$$k(G) = \max_{\Gamma}\{k(G^\Gamma) : \Gamma \text{ é um grafo 1-regular sobre } V(G)\}.$$

Sendo assim, temos o seguinte problema, equivalente da Mediana-c_4:

Problema Max-2/4-Ciclos(G): Dado um grafo G p-regular e p-aresta-colorido sobre $|V(G)| = 2n > 0$, encontrar um grafo 1-regular Γ sobre $V(G)$ tal que $k(G) = k(G^\Gamma)$.

A Figura 3.3 mostra um exemplo deste problema. Tendo em vista que o problema da mediana DCJ é NP-difícil [13], vamos nos concentrar na versão Max-2/4-Ciclos e na forma simplificada, onde $p = 3$.



(a) (b)

Figura 3.3: (a) Três emparelhamentos perfeitos $\Pi_1 = \{(1^h4^h, 4^t3^h, 3^t1^t, 2^t2^h\}, \Pi_2 = \{1^t4^h, 4^t3^h, 3^t2^h, 2^t1^h\}$ e $\Pi_3 = \{1^t1^h, 2^t2^h, 3^t3^h, 4^t4^h\}$ representados em um grafo G. (b) Uma solução ótima $\Gamma = \{(1\ 3\ 4), (2)\}$ com 7 ciclos: 5 ciclos de comprimento 2 (dois ciclos 1-coloridos (vermelho), dois 2-coloridos (azul), e um 3-colorido (verde)) mais dois ciclos de comprimento 4 (um ciclo 1-colorido (vermelho) e um 2-colorido (azul)). Portanto, $K^\star(\mathbf{\Pi}) = K(\mathbf{\Pi}, \Gamma) = 3 \cdot 4 - 7 = 5$.

3.3 Limitantes

Nesta seção apresentamos limitantes inferiores e superiores para o problema Max-2/4-Ciclos, tal que a instância G é um grafo 3-aresta-colorido com $2n$ vértices.

Seja G um grafo e $v, w \in V(G)$. O número de arestas entre v e w é a multiplicidade de v, w, que é denotado por $\mu(v, w)$. Seja e uma aresta que é incidente a v e w, então escrevemos $e = vw$ e dizemos que e é uma aresta de multiplicidade $\mu(v, w)$. Uma aresta de multiplicidade 1 é chamada de **aresta simples** e uma aresta de multiplicidade no mínimo 2 é chamada de **multiaresta**.

Um grafo 3-regular é chamado de **grafo cúbico**. Para o nosso estudo, a representação de adjacências dos três genomas no grafo de *breakpoint* resulta em um grafo cúbico 3-aresta-colorido, pois o conjunto de arestas pode ser particionado em três emparelhamentos perfeitos, que também são chamados de **classes de cores** de G.

Seja G um grafo cúbico 3-aresta-colorido com $2n$ vértices e Γ um grafo 1-regular sobre $V(G)$ com arestas de cor $p + 1$, sendo $p = 3$. Se $G^\Gamma = G \cup \Gamma$ tem n 2-ciclos i-coloridos, então $k(G) \geq n = \frac{|V(G)|}{2}$.

Note que toda aresta de G está contida em no máximo um ciclo i-colorido de G^Γ, **2-ciclo i-colorido** contém precisamente uma aresta de G e **4-ciclo i-colorido** de G^Γ contém precisamente duas arestas de G.

Proposição 1. *Seja G um grafo cúbico conexo 3-aresta-colorido. Então $k(G) \leq \frac{3}{2}|V(G)|$. Além disso, $k(G) = \frac{3}{2}|V(G)|$ se, e somente se, $G = K_2^3$ é o único grafo cúbico com dois vértices.*

Demonstração. No primeiro lado, se $G = K_2^3$, isso significa que o único par de vértices é escolhido para fazer parte de Γ, dessa forma $k(G) = 3$. Para outra direção, escolha Γ tal que $k(G^\Gamma) = k(G)$. Se $k(G) = \frac{3}{2}|V(G)|$, então segue das observações acima que G^Γ tem somente 2-ciclos i-coloridos. Portanto, $G = K_2^3$. □

Teorema 2. *Seja m um inteiro positivo e seja $G \neq K_2^3$ um grafo cúbico conexo 3-aresta-colorido. Se G tem m multiarestas, então $k(G) \leq |V(G)| + \lfloor \frac{m}{2} \rfloor$. Além disso, o limitante é preciso.*

Demonstração. Como K_2^3, pela proposição 1, é o único grafo cúbico conexo que contém uma aresta e com $\mu(e) = 3$, segue que G tem m arestas de multiplicidade 2. Seja $|V(G)| = 2n > 2$.

Seja Γ um grafo 1-regular sobre $V(G)$. Para $i \in \{1, 2\}$ seja $E_i(G) = \{e : e \in E(G) \text{ e } \mu(e) = i\}$, e $m_i = |E_i(G) \cap \Gamma|$. Como $m_1 + m_2$ é o número de arestas em um subconjunto de Γ, segue que $m_1 + m_2 \leq n$, sendo n o número de marcadores. Além disso, $m_2 \leq m$.

O grafo G^Γ tem $2m_2 + m_1$ 2-ciclos i-coloridos que cobrem $2m_2 + m_1$ arestas de G. Como todo 4-ciclo i-colorido contém precisamente duas arestas de G e cada aresta de G contém, no máximo, um ciclo i-colorido, segue que há, no máximo, $\frac{1}{2}(3n - (2m_2 + m_1))$ 4-ciclos i-coloridos. Portanto,

$$\begin{aligned} k(G^\Gamma) &\leq 2m_2 + m_1 + \tfrac{1}{2}(3n - (2m_2 + m_1)) \\ &= \tfrac{1}{2}(m_1 + m_2) + \tfrac{1}{2}m_2 + \tfrac{3}{2}n \leq 2n + \tfrac{1}{2}m_2. \end{aligned}$$

Como $m_2 \leq m$ e $k(G^\Gamma)$ é um inteiro, segue que $k(G^\Gamma) \leq 2n + \lfloor \frac{m}{2} \rfloor = |V(G)| + \lfloor \frac{m}{2} \rfloor$. E como Γ foi escolhido arbitrariamente, a primeira afirmação do teorema é provada.

Para verificar o limitante do teorema 2, veja a Figura 3.4. □

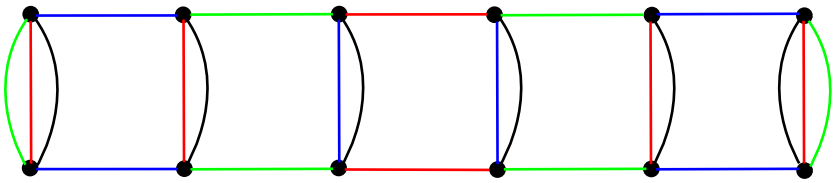


Figura 3.4: Uma escada linear, ou seja, um grafo cúbico conexo 3-aresta-colorível G com $|V(G)| = 2n = 12$, arestas com cores 1 (vermelho), 2 (azul) e 3 (verde), $m = 2$ multiarestas, e um grafo 1-regular Γ (arestas pretas) tal que $k(G^\Gamma) = k(G) = 13 = 2n + \lfloor \frac{m}{2} \rfloor$.

Sejam $n \geq 2$ e P_n um caminho nos vértices ordenados $v_1, \ldots, v_n$ e seja $P'_n = v'_1, \ldots, v'_n$ uma cópia de P_n. Seja $\mathcal{L}(n)$ um grafo cúbico obtido de P_n e P'_n adicionando as arestas $v_i v'_i$ para $i \in \{1, \ldots, n\}$ e duplicando as arestas $v_1 v'_1$ e $v_n v'_n$. Chamamos o grafo de **escada linear** se for isomorfo a $\mathcal{L}(n)$ para um inteiro $n \geq 2$. Observe a Figura 3.4 como exemplo.

Seja G_4^3 o grafo cúbico conexo em 4 vértices que possui multiarestas. Note que $k(G_4^3) = 5 = |V(G_4^3)| + \lfloor \frac{m}{2} \rfloor$. Esse grafo pode ser caracterizado por sua mediana-c_4.

Proposição 3. *Seja G um grafo cúbico conexo 3-aresta-colorido. Então $k(G) = \frac{5}{4}|V(G)|$ se, e somente se, $G = G_4^3$.*

Demonstração. Seja G um grafo com $2n$ vértices. Segue do Teorema 2 que a quantidade de multiarestas é igual a número de marcadores ($m = n$). Assim, as arestas de multiplicidade 2 formam um emparelhamento perfeito em G. Dessa forma, G é obtido a partir de um ciclo uniforme dobrando toda segunda aresta. Agora, é fácil observar que G_4^3 é o único grafo com $k(G) = \frac{5}{2}n$. A outra direção é trivial. □

A Figura 3.5 ilustra um exemplo da proposição 3.

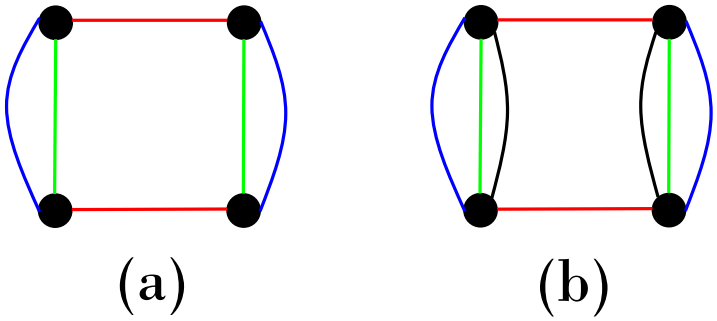


Figura 3.5: (a) Grafo G_4^3 com $m = n = 2$. (b) Grafo G^Γ onde $k(G) = \frac{5}{4}|V(G)| = 5$.

Corolário 4. *Seja G um grafo cúbico conexo 3-aresta-colorido. Se $|V(G)| > 4$, então $k(G) < \frac{5}{4}|V(G)|$.*

Para grafos simples podemos provar alguns limitantes melhores. Para $m \in \{0, 2\}$, todos os grafos atingem o limitante do Teorema 2. Talvez seja verdade que a família da Figura 3.4 caracteriza os grafos com máximo $k(G)$ e duas multiarestas. Se um grafo cúbico conexo G tem mais que duas multiarestas, então $k(G) < 2n + \lfloor \frac{m}{2} \rfloor$.

Seja C_n um ciclo com vértices ordenados $v_1, \ldots, v_n$ e C'_n uma cópia de C_n para algum inteiro $n \geq 3$. Seja $L_1(n)$ um grafo cúbico obtido de C_n e C'_n adicionando as arestas $v_i v'_i$ para $i \in \{1, \ldots, n\}$ e $L_2(n)$ um grafo cúbico obtido de $L_1(n) - \{v_n v_1, v'_n v'_1\}$ adicionando as arestas $v_n v'_1$ e $v'_n v_1$. Veja a Figura 3.6. O grafo $L_2(n)$ é também chamado de **escada de Möbius**.

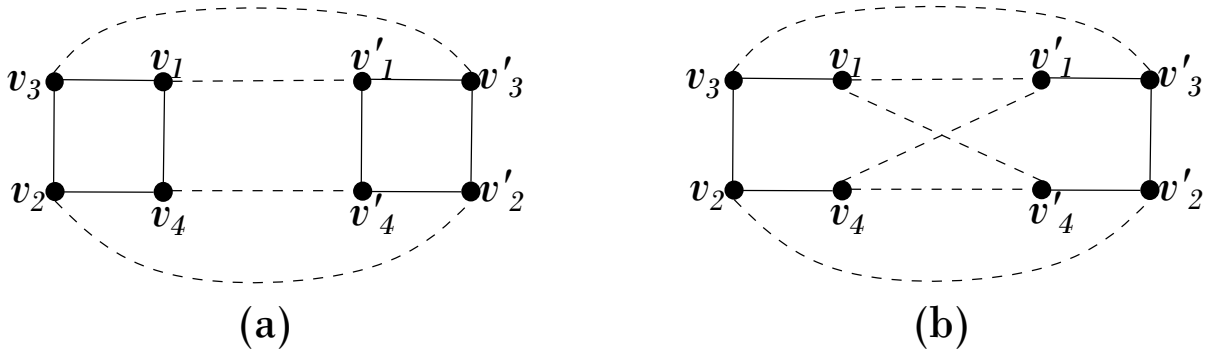


Figura 3.6: (a) $L_1(n)$ e (b) $L_2(n)$, sendo $n = 4$.

Chamamos o grafo de **escada circular** se for isomorfo ao K_4 ou ao $L_1(n)$ ou $L_2(n)$ para um inteiro $n \geq 3$.

Teorema 5. *Se G é um grafo cúbico conexo simples 3-aresta-colorido, então $k(G) \leq |V(G)|$. Além disso, $k(G) = |V(G)|$ se, e somente se, G é uma escada circular.*

Demonstração. A primeira parte segue do Teorema 2, sendo G um grafo simples. Se G é uma escada circular, então $k(G^\Gamma) = 2n = |V(G)|$ para $\Gamma = \{v_i v'_i : i \in \{1, \ldots, n\}\}$.

Seja Γ um grafo 1-regular com $2n$ vértices tal que $k(G^\Gamma) = k(G)$. Para $j \in \{2, 4\}$, seja c_j o número de j-ciclos i-coloridos. Temos $k(G^\Gamma) = c_2 + c_4 = 2n$. Consequentemente, $c_4 = \frac{1}{2}(3n - c_2)$

e além disso, $c_2 = c_4 = n$. Dessa forma, Γ é um emparelhamento perfeito de G. Seja $e \in \Gamma$ com $e = vw$, e sejam v_1, v_2 e w_1, w_2 os outros dois vizinhos de v e w, respectivamente. As arestas vv_i e ww_i não podem ser um 2-ciclo porque G é um grafo simples e Γ um emparelhamento. Assim, cada um deles está em um 4-ciclo i-colorido, e além disso, Γ induz a um emparelhamento perfeito em $G[\{v, v_1, v_2, w, w_1, w_2\}]$. Digamos que $v_1w_1, v_2w_2 \in \Gamma$. Segue que $v_1w_1, v_2w_2 \in E(G)$, sendo Γ um emparelhamento perfeito de G. Se $|V(G)| = 2n = 4$, então $G = K_4$. Se $|V(G)| = 2n > 4$, então G é isomorfo à escada circular com $2n$ vértices. □

A Figura 3.7 ilustra um exemplo do teorema 5.

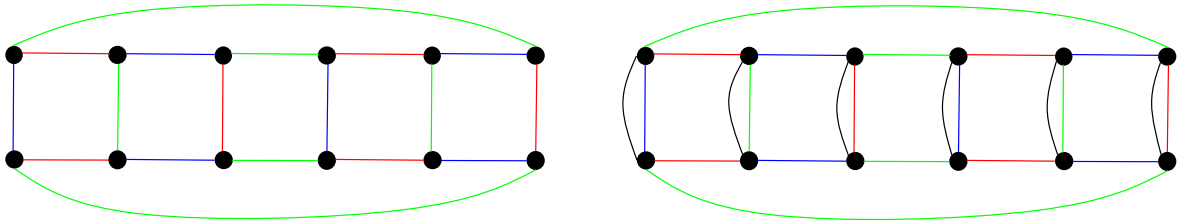


Figura 3.7: (a) escada circular G. (b) grafo G^Γ, onde $k(G) = 2n = 12$, sendo $n = 6$.

3.4 *Decomposers*

Uma estratégia conhecida para construir soluções para o problema da mediana é decompor $BG_x(\mathbf{\Pi})$ em pequenas partes e então identificar soluções parciais. Essas são posteriormente integradas à mediana completa [15, 14, 13, 11, 4]. A seguir, fazemos o uso da notação formal de [15] para caracterizar tais soluções parciais para o problema Max-2/4-Ciclos.

Seja $G = BG_x(\mathbf{\Pi})$ um grafo cúbico 3-aresta-colorido e Γ um grafo 1-regular sobre $V(G)$ tais que $k(G) = k(G^\Gamma)$. Para qualquer subgrafo induzido H de G, Γ é um ***H-crossing*** se, e somente se, contém uma aresta $uv \in \Gamma$ tal que $|\{u, v\} \cap V(H)| = 1$. Por outro lado, o subgrafo induzido H de G é um ***decomposer*** se, e somente se, existe um grafo 1-regular Γ com conjunto de vértices $V(G)$ tal que $k(G) = k(G^\Gamma)$ e Γ não é um *H-crossing*. H é um ***decomposer* forte** se cada grafo 1-regular Γ com $k(G) = k(G^\Gamma)$ não é um *H-crossing*. Das proposições 1, 3 e do teorema 5 seguem diretamente:

Corolário 6. *K_2^3, G_4^3 e toda escada linear e circular são decomposers fortes.*

Além disso, seja K_2^2 um grafo de dois vértices conectados por duas arestas paralelas de cores distintas.

Proposição 7. *K_2^2 é um decomposer forte.*

Demonstração. Seja K_2^2 um subgrafo induzido de um grafo cúbico 3-aresta-colorido $G = BG_x(\mathbf{\Pi})$, $\{u, v\} = V(K_2^2)$, e $u'u, v'v \in G - \{uv\}$. Observe $u' \neq v'$. Assuma por contradição que para algum Γ tal que $k(G) = k(G^\Gamma)$, $uv \notin \Gamma$. Então $u'u$, $v'v$ devem estar em Γ. Existem dois casos: se $u'v' \in G$, então o subgrafo induzido por vértices u, u', v, v' compartilham três ciclos com Γ, mas quatro ciclos com $\Gamma' = \Gamma - \{u'u, v'v\} + \{uv, u'v'\}$. Caso contrário, o

subgrafo induzido por vértices u, u', v, v' compartilham dois ciclos com Γ, porém três ciclos com $\Gamma' = \Gamma - \{u'u, v'v\} + \{uv, u'v'\}$. □

A Figura 3.8 ilustra um exemplo para a proposição 7.

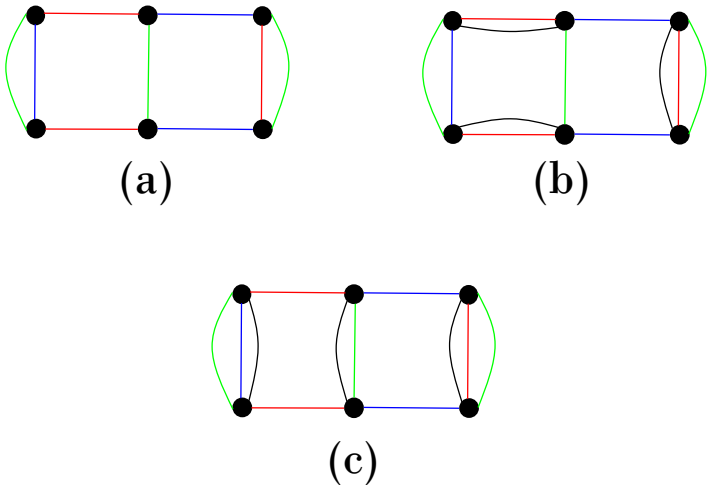


Figura 3.8: (a) Grafo cúbico 3-aresta-colorido G. (b) G^{Γ} com $k(G) = 5$ e (c) $G^{\Gamma'}$ com $k(G) = 7$.

Decomposers de problemas relacionados

Ao analisar o problema Mediana-c_4, uma questão simples é se alguns *decomposers* de problemas relacionados, a mediana SCJ e DCJ, também são *decomposers* para Mediana-c_4. Para a mediana SCJ, sabe-se o seguinte:

Proposição 8. *K_2^2 e K_2^3 são decomposers fortes da mediana de breakpoint de três [11].*

Além disso, é fácil observar que todo componente conexo em $BG_x(\mathbf{\Pi})$ é um *decomposer* da mediana SCJ.

Subgrafos adequados [15] são da família dos *decomposers* da mediana DCJ: o subgrafo $H \subset G$ é um subgrafo adequado se $k(H) \geq \frac{3}{4}|V(G)|$.

Proposição 9. *Subgrafos adequados não são decomposers do problema* Mediana-c_4.

Demonstração. Ciclos de quatro vértices $v_1, \ldots, v_4$ são subgrafos adequados [15]. Figura 3.9 ilustra um contra exemplo onde todas as medianas-c_4 são *H-crossings* para o ciclo H em destaque. □

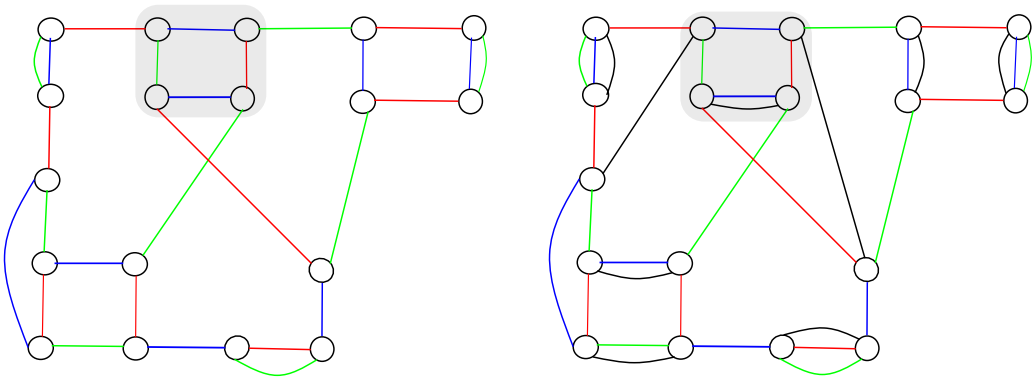


Figura 3.9: (a) Grafo cúbico 3-aresta-colorido G com subgrafo adequado embutido destacado em cinza e (b) grafo G^{Γ} com $k(G) = k(G^{\Gamma}) = 15$.

Capítulo 4

Algoritmos

Neste capítulo apresentamos dois algoritmos exatos em programação linear inteira (PLI) e três heurísticas combinatórias chamados Ciclos bicoloridos, Encurtamento de adjacências e Pontuação de arestas. Além disso, apresentamos quatro algoritmos que são adapatações da heurística Pontuação das arestas: Vizinhos adjacentes, Maior potencial de ciclo e menor pontuação, Sorteio das arestas e Arestas 4-ciclos.

4.1 Programação linear inteira

Nosso algoritmo exato PLI traduz a fórmula de minimização do problema da Mediana-c_4 de uma maneira simples, veja o algoritmo 1.

Suponha que recebemos um conjunto de três emparelhamentos perfeitos $\mathbf{\Pi} = \{\Pi_1, \Pi_2, \Pi_3\}$ em um grafo G. Em seguida, verificamos se uma aresta arbitrária forma um 2-ciclo i-colorido ou se duas arestas arbitrárias formam um 4-ciclo i-colorido em G, para cada possível aresta conformada por um par em $V(G)$. Finalmente, maximizamos essas quantidades para obter uma solução do problema Max-2/4-Ciclos.

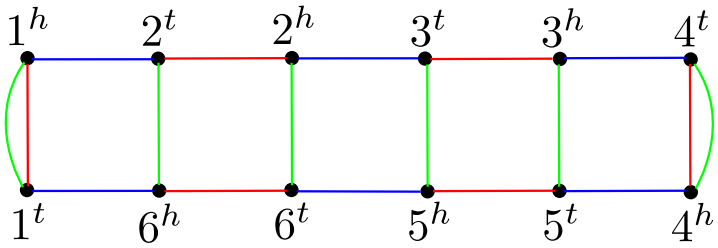
Algoritmo 1 PLI para computar a mediana c_4

min			$3n - \sum_{\substack{\pi \in \Pi_i \\ \Pi_i \in \mathbf{\Pi}}} c^{\pi}_{2,i} - \sum_{\substack{\pi,\sigma \in \Pi_i \\ \Pi_i \in \mathbf{\Pi}}} c^{\pi,\sigma}_{4,i}$		
sujeito a	$\sum_{\substack{v \in V(G) \\ v \neq u}} uv$	$=$	1	$\forall\, u \in V(G)$	(R.01)
	$c^{\pi}_{2,i}$	$=$	π	$\forall\, \pi = uv \in \Pi_i, \forall\, \Pi_i \in \mathbf{\Pi}$	(R.02)
	$2c^{\pi,\sigma}_{4,i}$	$\leq$	$ux + uy +$ $vx + vy$	$\left.\begin{array}{l} \forall\, \pi = uv \\ \forall\, \sigma = xy \end{array}\right\} \in \Pi_i, \forall\, \Pi_i \in \mathbf{\Pi}$	(R.03)
e	uv	$\in$	$\{0,1\}$	$\forall\, u, v \in V(G), u \neq v$	(D.01)
	$c^{\pi}_{2,i}$	$\in$	$\{0,1\}$	$\forall\, \pi = uv \in \Pi_i, \forall\, \Pi_i \in \mathbf{\Pi}$	(D.02)
	$c^{\pi,\sigma}_{4,i}$	$\in$	$\{0,1\}$	$\left.\begin{array}{l} \forall\, \pi = uv \\ \forall\, \sigma = xy \end{array}\right\} \in \Pi_i, \forall\, \Pi_i \in \mathbf{\Pi}$	(D.03)

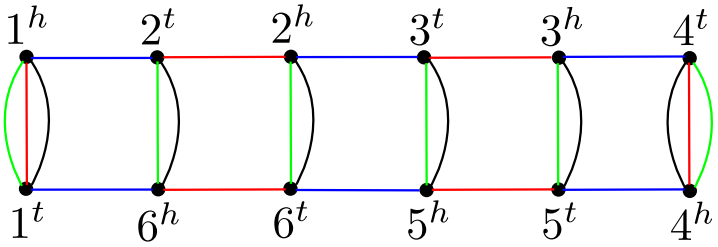
Impomos que uma, e somente uma, aresta na solução é escolhida para cada possível extremi-

dade u em $V(G)$ (restrição (R.01) e variável binária (D.01)). Se uma aresta $\pi = uv$ do genoma Π_i é escolhida para as extremidades $u, v \in V(G)$, então a variável binária $c^{\pi}_{2,i}$ recebe 1. Caso contrário, $c^{\pi}_{2,i}$ recebe 0 (restrição (R.02), variável binária (D.02)). Além disso, se tivermos duas arestas $\pi = uv$ e $\sigma = xy$ em $V(G)$ e o par de arestas distintas é escolhido para as extremidades u, v, x, y em $V(G)$, então a variável binária $c^{\pi,\sigma}_{4,i}$ recebe 1. Caso contrário, $c^{\pi,\sigma}_{4,i}$ recebe 0 (restrição (R.03), variável binária (D.03)). Finalmente, para todos os emparelhamentos Π_i, $1 \leq i \leq 3$, maximizamos as quantidades $c^{\pi}_{2,i}$ e $c^{\pi,\sigma}_{4,i}$ para todas as possíveis arestas π e σ para os vértices de $V(G)$, o que corresponde à função objetivo da PLI.

Observe que temos $O(n^2)$ restrições e variáveis binárias no algoritmo 1. A figura 4.1 mostra um exemplo da primeira versão do algoritmo.



(a)



(b)

Figura 4.1: (a) Emparelhamentos perfeitos $\Pi_1 = \{1^t1^h, 2^t2^h, 3^t3^h, 4^t4^h, 3^t5^h, 6^t6^h\}$, $\Pi_2 = \{1^h2^t, 2^h3^t, 3^h4^t, 4^h5^t, 5^h6^h, 6^h1^t\}$ e $\Pi_3 = \{1^t1^h, 2^h6^t, 6^h2^t, 3^h5^t, 5^h3^t, 4^t4^h\}$ representados em um grafo cúbico 3-aresta-colorido G. (b) Grafo G^{Γ} tal que $\Gamma = \{1^t1^h, 2^h6^t, 6^h2^t, 3^h5^t, 5^h3^t, 4^t4^h\}$ contém as arestas que maximizam a quantidade de 2- e 4-ciclos em G^{Γ}.

Com o objetivo de acelerar a busca pela solução ótima, foi proposta uma segunda versão do algoritmo exato PLI, o qual é descrito no algoritmo 2.

Considere o grafo $K_{2n} = (V, E)$. Seja $\Pi_1 = (V_1, E_1)$ tal que $V_1 = V$ e $d(v) = 1$ para todo v em V_1. Ou seja, Π_1 é um emparelhamento perfeito em K_{2n}. De igual modo, sejam $\Pi_2 = (V_2, E_2)$ e $\Pi_3 = (V_3, E_3)$. Além disso, seja $c_{\pi} = \sum_{i \in \{1,2,3\}} |E_i \cap \{\pi\}|$ a quantidade de 2-ciclos que a aresta π contém e seja $c_{\pi,\sigma} = \sum_{i \in \{1,2,3\}} c_4(\pi, \sigma, E_i)$ a quantidade de 4-ciclos que as arestas distintas π e σ contêm juntas, sendo $\pi, \sigma \in E$. Seja $F = \{\{\pi, \sigma\} : \pi, \sigma \in E, \pi \neq \sigma \text{ e } c_{\pi,\sigma} > 0\}$ um conjunto de pares de arestas tais que π e σ estão em E e formam um 4-ciclo.

Algoritmo 2 Segunda versão do algoritmo exato PLI para computar a mediana c_4

min		$3n - \sum_{\pi \in E} c_\pi \; y_\pi - \sum_{\pi,\sigma \in E, \pi \neq \sigma} c_{\pi,\sigma} \; x_{\pi,\sigma}$			
sujeito a	$\sum_{\pi \sim v} y_\pi$	$=$	1	$\forall \; v \in V$	(R.01)
	$x_{\pi,\sigma}$	$\leq$	y_π	$\forall \; \{\pi, \sigma\} \in F$	(R.02)
	$x_{\pi,\sigma}$	$\leq$	y_σ		
e	y_π	$\in$	$\{0,1\}$	$\forall \; \pi \in E$	(D.01)
	$x_{\pi,\sigma}$	$\in$	$\{0,1\}$	$\forall \; \pi, \sigma \in E, \pi \neq \sigma$	(D.02)

Estabelecemos que uma, e somente uma, aresta na solução é escolhida para cada possível extremidade v em V_i e se a aresta π em E é escolhida, então a variável binária y_π recebe 1, caso contrário recebe 0 (restrição (R.01) e variável binária (D.01)). Se o par de arestas distintas π e σ é escolhido para as arestas que estão em E, então a variável binária $x_{\pi,\sigma}$ recebe 1. Caso contrário, $x_{\pi,\sigma}$ recebe 0 (restrição (R.02), variável binária (D.03)). Finalmente, maximizamos a quantidade de 2- e 4-ciclos com as variáveis c_π e $c_{\pi,\sigma}$, respectivamente, que correspondem à função objetivo da PLI.

Uma das diferenças entre as duas versões do algoritmo PLI está no fato de que as escolhas das arestas que formam 2-ciclos estão vinculadas ao contador c_π, quando uma aresta é escolhida para fazer parte da solução, conformando um 2-ciclo. A outra diferença está na escolha dos pares de arestas para a formação de 4-ciclos e na variável $c_{\pi,\sigma}$ é verificado se o par está em E_i com $i \in \{1, 2, 3\}$. A restrição é dividida em duas partes, pois isso acelera o processo na busca pela solução ótima.

Observe que também temos $O(n^2)$ restrições e variáveis binárias no algoritmo 2. A figura 4.2 mostra um exemplo da segunda versão do algoritmo.

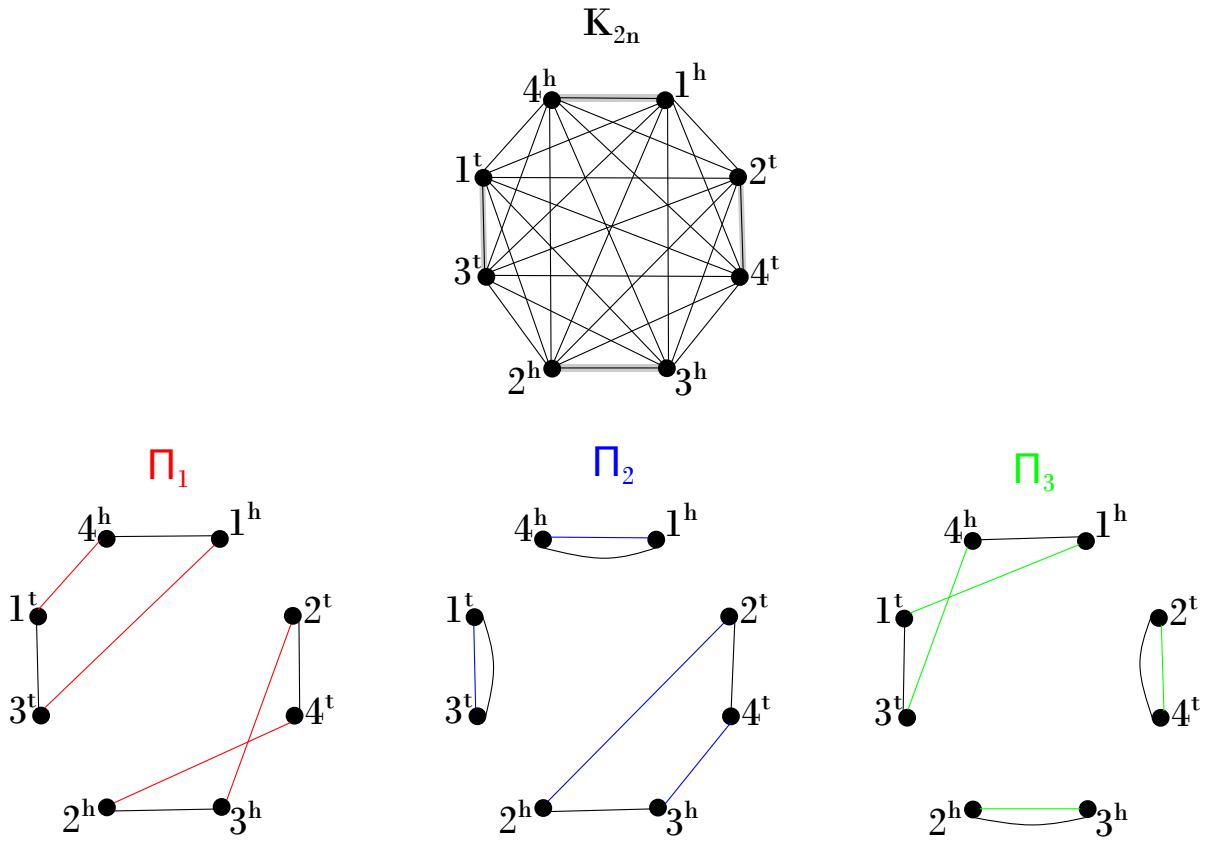


Figura 4.2: Grafo completo K_{2n} sendo $n = 4$. Conforme as arestas em K_{2n} são escolhidas, o número de 2- e 4-ciclos formados com as arestas do grafo cúbico 3-aresta-colorido G é computado.

4.2 Ciclos bicoloridos induzidos

A ideia básica do algoritmo 3 é a seguinte. Tendo um grafo cúbico 3-aresta-colorido G representando os emparelhamentos perfeitos $\mathbf{\Pi} = \{\Pi_1, \Pi_2, \Pi_3\}$ como entrada, tomamos dois dos emparelhamentos dados Π_i e Π_j de $\mathbf{\Pi}, i, j \in \{1, 2, 3\}, i \neq j$, e construímos um grafo induzido $H_{i,j} = G[\Pi_i \cup \Pi_j]$. Observe que $H_{i,j}$ é uma coleção de ciclos bicoloridos, com alternância de arestas i- e j-coloridas. Para cada ciclo bicolorido C em $H_{i,j}$, obtemos um conjunto de arestas que representa uma escada linear em C, adicionamos o emparelhamento restante à $H_{i,j}$ e computamos o número total de 2- e 4-ciclos obtido. Chamamos $\Gamma_{i,j}$ o conjunto dessas arestas. Repetimos o processo para cada par $i, j \in \{1, 2, 3\}, i \neq j$ e devolvemos o melhor dos três.

Na linha 2 do algoritmo 3, o grafo induzido $H_{i,j}$ é construído utilizando as arestas i e j-coloridas do grafo cúbico 3-aresta-colorido G. Em seguida, $\Gamma_{i,j}$ representa a mediana dos genomas i e j e é inicializada como um conjunto vazio de arestas. Na linha 4, cada ciclo C é representado por um conjunto de vértices $V(C) \subseteq V(G)$ e, para obtê-lo, é usado um procedimento que visita os vértices que não foram visitados no grafo $H_{i,j}$ partindo de qualquer vértice, finalizando a visitação no mesmo e assinalando-os como visitados.

Na linha 5, o procedimento para obter o conjunto de arestas A' é o seguinte: para cada ciclo C de tamanho w, uma escada linear A' é construída a partir de uma aresta $uv \in C$, ou seja, $A' = \{uv, u_1v_1, u_2v_2, u_3v_3, \ldots, u_lv_l\}$, com $l = |V(C)|/2$. Na linha 6, as arestas de A' que contêm

Algoritmo 3 Ciclos bicoloridos induzidos

Entrada: Grafo cúbico 3-aresta-colorido G obtido dos emparelhamentos perfeitos em $\mathbf{\Pi}$

Saída: Grafo 1-regular Γ sobre $V(G)$ tal que $k(\mathbf{\Pi}, \Gamma_{i,j})$ é máximo para qualquer par $i, j \in \{1, 2, 3\}, i \neq j$

1: **para cada** par $i, j \in \{1, 2, 3\}, i \neq j$ **faça**
2: Construa um grafo induzido $H_{i,j}$ pelos genomas Π_i, Π_j
3: $\Gamma_{i,j} \leftarrow \emptyset$
4: **para cada** ciclo C em $H_{i,j}$ **faça**
5: Obtenha um conjunto de arestas A' representando uma escada linear em C que maximize o número de 2- e 4-ciclos em G
6: $\Gamma_{i,j} \leftarrow \Gamma_{i,j} \cup A'$
7: **devolva** $\Gamma \leftarrow \arg\max\{k(\mathbf{\Pi}, \Gamma_{i,j}) : i, j \in \{1, 2, 3\}, i \neq j\}$

a maior quantidade de 2- e 4-ciclos farão parte de $\Gamma_{i,j}$. No final, após obter os emparelhamentos perfeitos para os três pares de emparelhamentos perfeitos de entrada distintos, é devolvido aquele que resulta na maior quantidade de 2- e 4-ciclos.

Algoritmo 3 pode ser implementado de tal forma que seu consumo de tempo seja $O(n^3)$ no tamanho da representação do grafo G. Na linha 1, o laço é executado $3 = \binom{3}{2}$ vezes e a construção do grafo induzido $H_{i,j}$ consume tempo $O(n)$. Na linha 4, o número máximo de ciclos é n, ou seja, o laço é executado no máximo n vezes. Na linha 5, antes de obter o conjunto de aresta A' que contém o maior número de 2- e 4-ciclos, os seguintes procedimentos são executados: a construção da escada linear partindo de uma aresta $uv \in C$ e a contagem de 2- e 4-ciclos da escada linear com o ciclo C juntamente com o terceiro emparelhamento Π_k, sendo $k \neq i \neq j$. A construção da escada linear consome tempo $O(V(C))$ e a contagem de 2- e 4-ciclos tem tempo $O(V(C)^2)$, já que há uma combinação dos pares de arestas da escada linear para calcular a quantidade de 4-ciclos. Ambos os procedimentos pertencem ao corpo de instruções de uma estrutura de repetição que é executada um número de vezes igual à metade do comprimento do ciclo C, ou seja, tem tempo $O(V(C))$. Dessa forma, o tempo total de execução do algoritmo 3 é $O(n^3)$, já que $|V(C)| = O(n)$.

Um exemplo ilustrando a execução do algoritmo 3 é apresentado na figura 4.3.

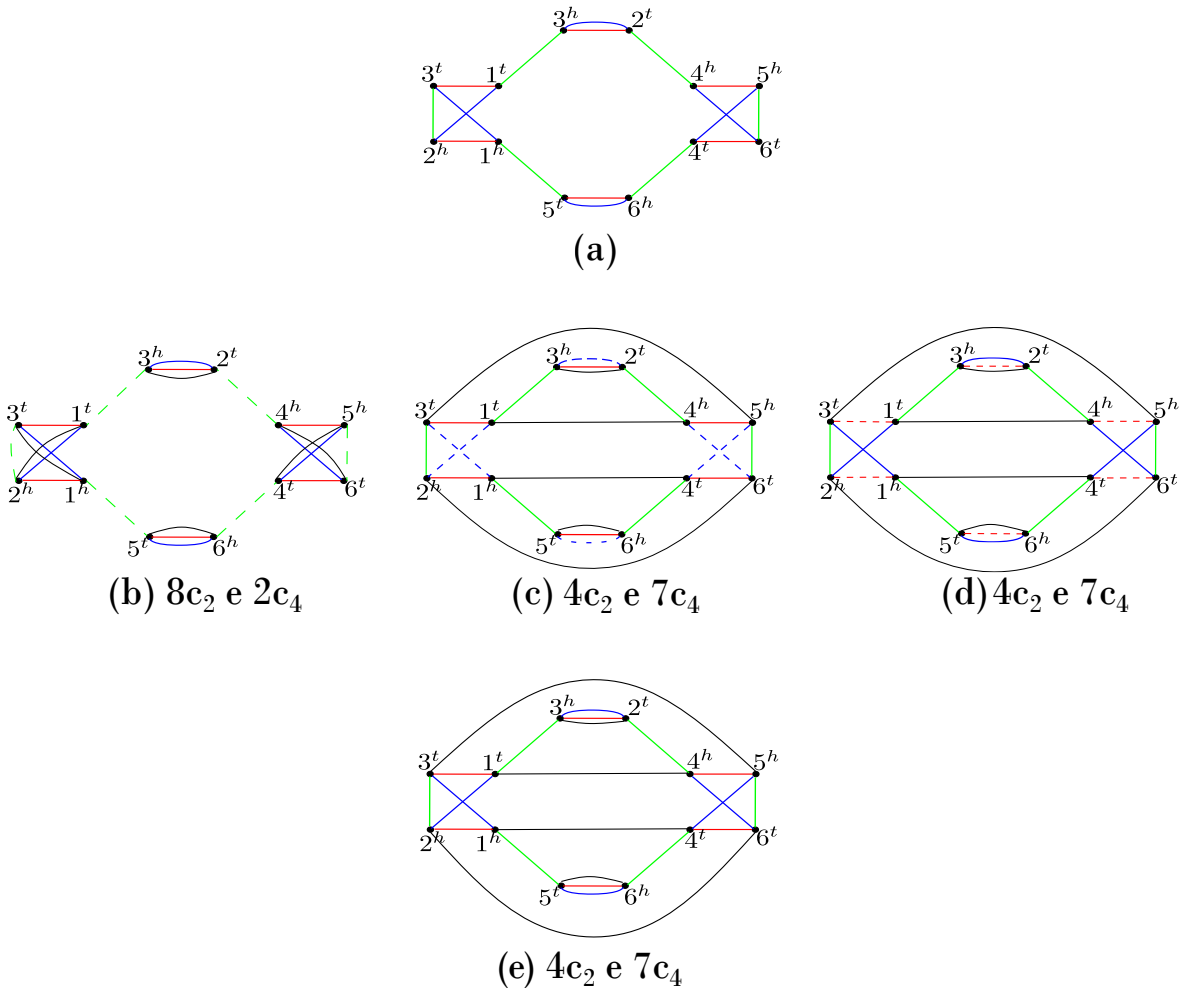


Figura 4.3: (a) Grafo G contendo os emparelhamentos perfeitos $\Pi_1 = \{(1^h2^h, 2^t3^h, 3^t1^t, 6^h5^t5^h4^h4^t6^t\}, \Pi_2 = \{1^h3^t, 3^h2^t, 2^h1^t, 4^h6^t, 6^h5^t, 5^h4^t\}$ e $\Pi_3 = \{2^h3^t, 3^h1^t, 1^h5^t, 5^h6^t, 6^h4^t, 4^h2^t\}$. Grafos induzidos (b) $H_{1,2}$, (c) $H_{1,3}$ e (d) $H_{2,3}$ contendo as arestas candidatas à mediana c_4. (e) O candidato que contém o número máximo de 2- e 4-ciclos é $\Gamma = \{1^h4^t, 4^h1^t, 6^h5^t, 5^h3^t, 3^h2^t, 2^h6^t\}$, o qual é a solução ótima.

4.3 Encurtamento de adjacências

Dado um grafo cúbico 3-aresta-colorido G representando os emparelhamentos perfeitos $\mathbf{\Pi} = \{\Pi_1, \Pi_2, \Pi_3\}$, o encurtamento de adjacências é uma estratégia gulosa que encontra um grafo 1-regular Γ sobre $V(G)$.

Seja G um grafo com $2n$ vértices e $3n$ arestas. Sejam u, v vértices de $V(G)$ e $uv \in E(G)$. Sejam u_i e v_i vértices em $V(G)$ tais que $u_i \neq v_i$, e uu_i e vv_i são arestas i-coloridas.

Para cada cor $i \in \{1, 2, 3\}$, seja $G' := G + u_iv_i - \{uv, uu_i, vv_i\}$. Chamamos a aresta uv de **adjacência encurtada** em G. O processo de encurtar remove três a seis arestas e adiciona zero a três arestas em G', dependendo de quantas arestas com extremidades u e v existem em G. Dessa forma, o número de arestas em G' é $3n-3$, ou seja, G' é um grafo cúbico 3-aresta-colorido já que a cada aresta encurtada, as arestas incidentes à u e v serão removidas e adicionadas à

u_i e v_i, fazendo com que cada vértice em $V(G')$ seja incidente a três arestas. O algoritmo 3 implementa recursivamente essa ideia.

Algoritmo 4 Encurtamento de adjacências

Entrada: Grafo cúbico 3-aresta-colorido G obtido dos emparelhamentos perfeitos em $\mathbf{\Pi}$

Saída: Grafo 1-regular Γ sobre $V(G)$ contendo as arestas escolhidas pelo critério do ciclo mais curto

1: **enquanto** existir aresta em G **faça**
2: Escolha aresta $uv \in E(G)$ de acordo com o critério do ciclo mais curto
3: Sejam u_i e v_i os vértices em $V(G)$ tais que $u_i \neq v_i$, e vv_i e uu_i são arestas i-coloridas
4: **devolva** uv mais o resultado da chamada recursiva do encurtamento de adjacências para $G + u_iv_i - uv$

Critério do ciclo mais curto

Para $h \geq 1$, seja G_h um grafo cúbico 3-aresta-colorido da h-ésima chamada recursiva do algoritmo 4. Note que cada aresta no grafo cúbico 3-aresta-colorido pertence a exatamente dois ciclos bicoloridos. Considere a quádrupla $\mathrm{vl}(e), \mathrm{rl}(e), \mathrm{sh}(e), \mathrm{lg}(e)$, para cada aresta $e = uv \in E(G_h)$ tal que $\mathrm{vl}(e) \in \{0, 1\}$ denota o valor de contribuição da aresta e; $\mathrm{rl}(e) =$ *verdadeiro* denota que a aresta está em G, e $\mathrm{rl}(e) =$ *falso* caso contrário; e $\mathrm{sh}(e)$ e $\mathrm{lg}(e)$ são os comprimentos de dois ciclos bicoloridos mais curto e mais longo, respectivamente, que contêm a aresta e, com $\mathrm{sh}(e) \leq \mathrm{lg}(e)$.

Inicialmente, no grafo $G = G_1$, $\mathrm{vl}(e) = 1$ e $\mathrm{rl}(e) =$ *verdadeiro* para cada aresta $e \in E(G)$. O algoritmo 4 escolhe uma aresta i-colorida $e \in E(G_h)$ de acordo com o seguinte. Suponha que $e = uv$ e u_i, v_i são vértices tais que $u_i \neq v_i$, $e_u = uu_i$ e $e_v = vv_i$ são arestas i-coloridas. Quando os vértices u e v são removidos de G_h então, para cada i, definimos uma aresta i-colorida $e_i = u_iv_i$, e fazemos $\mathrm{rl}(e_i) =$ *falso* e $\mathrm{vl}(e_i) = 1$ se $\mathrm{rl}(vv_i) = \mathrm{rl}(uu_i) =$ *verdadeiro*. Caso contrário, $\mathrm{vl}(e_i) = 0$. Note que a remoção da aresta uv implica que a quádrupla de todas as arestas restantes, de ciclos aos quais essas arestas pertencem, deverão ser atualizadas. Isto significa que, em cada chamada recursiva, $O(n)$ arestas devem ter seus atributos atualizados.

Considerando a quádrupla $\mathrm{vl}(e), \mathrm{rl}(e), \mathrm{sh}(e), \mathrm{lg}(e)$, definimos uma ordem total $\preceq$ no conjunto das arestas de G_h. Dadas duas arestas e_1 e e_2, dizemos que $e_1 \preceq e_2$ se, e somente se, uma das condições a seguir é satisfeita:

1. $\mathrm{vl}(e_1) > \mathrm{vl}(e_2)$, ou
2. $\mathrm{vl}(e_1) = \mathrm{vl}(e_2), \mathrm{rl}(e_1) =$ *verdadeiro* e $\mathrm{rl}(e_2) =$ *falso*, ou
3. $\mathrm{vl}(e_1) = \mathrm{vl}(e_2), \mathrm{rl}(e_1) = \mathrm{rl}(e_2), \mathrm{sh}(e_1) < \mathrm{sh}(e_2)$, ou
4. $\mathrm{vl}(e_1) = \mathrm{vl}(e_2), \mathrm{rl}(e_1) = \mathrm{rl}(e_2), \mathrm{sh}(e_1) = \mathrm{sh}(e_2)$ e $\mathrm{lg}(e_1) \leq \mathrm{lg}(e_2)$.

Dizemos que $e \in E(G_h)$ é uma **aresta ótima** para o critério do ciclo mais curto se $e \preceq g$ para cada $g \in E(G_h)$. Em cada chamada recursiva do algoritmo 4, uma aresta ótima é escolhida.

Finalmente observe que, em cada chamada recursiva, temos de encontrar uma aresta $e = uv$ de acordo com o critério do ciclo mais curto, o que consome tempo $O(n)$, e temos que remover u e v de G_h, o que consome tempo $O(1)$. Então, temos que atualizar a quádrupla de todas as

arestas em ciclos envolvidos nesta operação e isso pode ser realizado em tempo $O(n)$. Assim, o algoritmo 4 consome tempo $O(n)$ em cada chamada recursiva e, portanto, seu tempo de execução é $O(n^2)$.

Um exemplo mostrando a execução do algoritmo 4 é apresentado na figura 4.4.

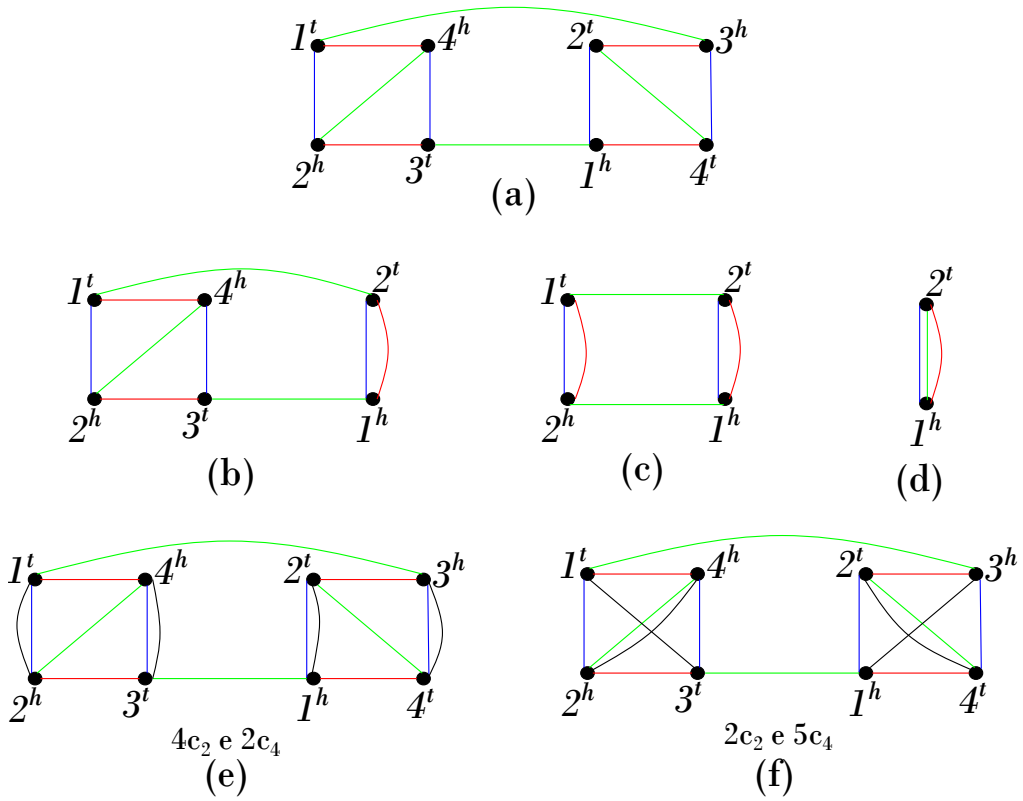


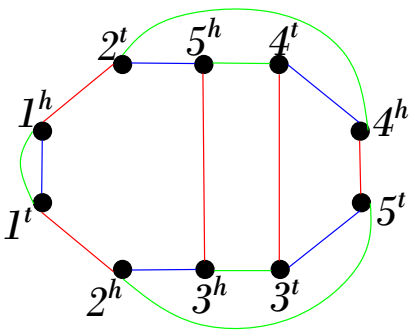
Figura 4.4: (a) Um grafo G e seus emparelhamentos perfeitos $\Pi_1 = \{2^h3^t, 3^h2^t, 1^h4^t, 4^h1^t\}$, $\Pi_2 = \{1^h2^t, 2^h1^t, 4^h3^t, 3^h4^t\}$ e $\Pi_3 = \{3^h1^t, 1^h3^t, 2^h4^h, 4^t2^t\}$. De acordo com os critérios, as arestas (b) 3^h4^t, (c) 3^t4^h, (d) 1^t2^h e (e) 1^h2^t foram encurtadas, resultando em $\Gamma = \{1^h2^t, 2^h1^t, 3^h4^t, 4^h3^t\}$. No entanto, para este exemplo, a solução ótima é $\Gamma^* = \{2^h4^h, 4^t2^t, 1^h3^h, 3^t1^t\}$ que contabiliza 7 ciclos.

4.4 Pontuação das arestas

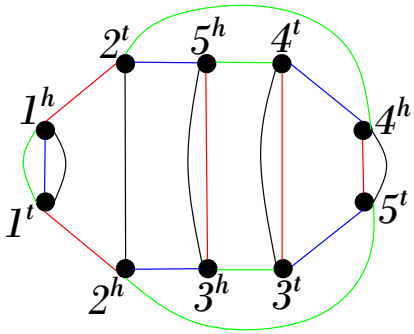
Seja G um grafo cúbico 3-aresta-colorido com $2n$ vértices representando o conjunto de emparelhamentos perfeitos $\mathbf{\Pi} = \{\Pi_1, \Pi_2, \Pi_3\}$. Uma aresta $e \in G$ é **confiável** se a multiplicidade da aresta e é maior ou igual a dois ($\mu(e) \geq 2$). Seja $\mathcal{R}$ o conjunto de arestas confiáveis.

Seja Γ um grafo 1-regular sobre $V(G)$. Defina a **pontuação s de uma aresta** uv em Γ como $s(uv) = t + \frac{1}{2}f$, onde t é o número de 2-ciclos i-coloridos que uv está contida e f é o número de 4-ciclos i-coloridos que uv está contida, com $i \in \{1, 2, 3\}$. Note que $0 \leq t + f \leq 3$. Seja $s(\Gamma) := \sum_{uv \in \Gamma} s(uv)$ e observe que $s(\Gamma) = k(G^\Gamma)$. As duas arestas em Γ de um 4-ciclo i-colorido em G^Γ são chamadas de **arestas irmãs**. A figura 4.5(b) mostra G^Γ tal que $s(1^t1^h) = \frac{5}{2}$, $s(2^t2^h) = \frac{3}{2}$ e $s(3^h5^h) = s(3^t4^t) = s(4^h5^t) = 2$.

Em seguida, para cada uv em Γ, defina seu **potencial de ciclo** λ em G^Γ como $\lambda(uv) =$



(a)



(b)

aresta e	s(e)	λ(e)
1^t1^h	2.5	0.0
2^t2^h	1.5	0.0
3^h5^h	2.0	0.0
3^t4^t	2.0	0.0
4^h5^t	2.5	0.0
Total	10.5	0.0

Figura 4.5: (a) Grafo G representando os genomas $\Pi_1 = \{(1\ 2), (3\ \overline{5}\ \overline{4})\}$, $\Pi_2 = \{(1), (2\ \overline{3}\ 5), (4)\}$ e $\Pi_3 = \{(1), (2\ 5\ 4), (3)\}$. (b) Grafo G^Γ, sendo $\Gamma = \{(1), (2), (3\ \overline{5}\ \overline{4})\}$.

$\frac{1}{2}(\mu(uv)+3)-s(uv)$. O potencial de ciclo $\lambda(uv)$ de uma aresta uv em Γ representa a possibilidade de envolvimento de uv em outros 2- e 4-ciclos em G^Γ. Referindo-se novamente a G^Γ na figura 4.5 (b), todas as arestas uv em Γ têm $\lambda(uv) = 0$.

O algoritmo 5 inicia escolhendo arestas confiáveis e arestas restantes aleatórias para fazer parte da solução inicial, obtendo o grafo 1-regular Γ sobre $V(G)$. Em seguida, é computada a pontuação e o potencial de ciclo para cada aresta em Γ. O próximo passo é tentar incrementar a pontuação das arestas e decrementar o potencial de ciclo de um pequeno subconjunto de arestas, através de alterações locais.

Seja uv uma aresta em Γ tal que $\lambda(uv) > 0$. Para $i \in \{1, 2, 3\}$, há pelo menos um par de arestas i-coloridas em G, digamos uu_1 e vv_1, tal que $u_1v_1 \notin \Gamma$. Como Γ é um emparelhamento perfeito, u_1 e v_1 são vértices saturados e sejam u_1u_2 e v_1v_2 as arestas em Γ. Agora, o algoritmo remove u_1u_2, v_1v_2 e adiciona u_1v_1, u_2v_2 em Γ e a pontuação e o potencial de ciclo da aresta uv devem ser alterados. Adicionalmente, a pontuação e o potencial de ciclo das novas arestas u_1v_1 e u_2v_2 devem ser computados.

O algoritmo executa essa operação se, e somente se, u_1u_2 e v_1v_2 não são arestas confiáveis. O processo se repete enquanto a soma das pontuações de todas as arestas aumenta de um passo para o outro e há uma aresta com potencial de ciclo positivo.

Algoritmo 5 Pontuação das arestas

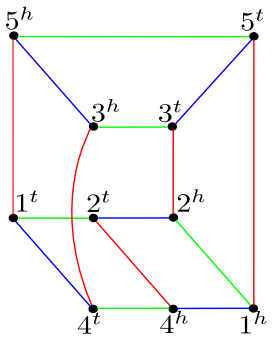
Entrada: Grafo cúbico 3-aresta-colorido G obtido dos emparelhamentos perfeitos em $\mathbf{\Pi}$

Saída: Grafo 1-regular Γ sobre $V(G)$ tal que $s(\Gamma)$ é máximo considerando fixas as arestas confiáveis

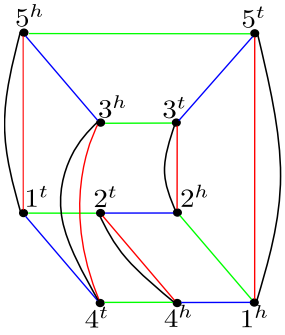
1: Seja $\mathcal{R}$ o conjunto das arestas confiáveis G
2: Seja Γ um emparelhamento perfeito em G formado por $\mathcal{R}$ e arestas restantes de G de forma arbitrária
3: Compute $s(uv), \lambda(uv)$ para cada aresta uv em Γ
4: **se** existe alguma aresta uv em Γ tal que $\lambda(uv) > 0$ **então**
5: Sejam uu_1, vv_1 arestas i-coloridas, $u_1v_1 \notin \Gamma$, $i \in \{1, 2, 3\}$
6: Sejam u_1u_2, v_1v_2 arestas em Γ
7: **se** $u_1u_2, v_1v_2 \notin \mathcal{R}$ **então**
8: $\Gamma = \Gamma + \{u_1v_1, u_2v_2\} - \{u_1u_2, v_1v_2\}$
9: Atualize $s(uv), \lambda(uv)$
10: Atualize s,λ para arestas irmãs de u_1u_2,v_1v_2,u_1v_1,u_2v_2
11: Compute $s(u_1v_1), \lambda(u_1v_1)$ e $s(u_2v_2), \lambda(u_2v_2)$
12: Repita as linhas 4–11 enquanto $s(\Gamma)$ pode ser incrementado sem remover as arestas em $\mathcal{R}$
13: **devolva** Γ

Observe que a busca na linha 4, para cada aresta em G, consome tempo $O(n)$ e cada linha de 5 a 11 pode ser executada em tempo constante. Além disso, $s(\Gamma)$ pode ser incrementado $O(n)$ vezes, o que significa que o tempo de execução do algoritmo 5 é $O(n^2)$.

Um exemplo mostrando a execução do algoritmo 5 é apresentado na figura 4.6.

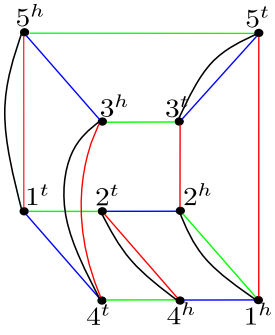


(a)



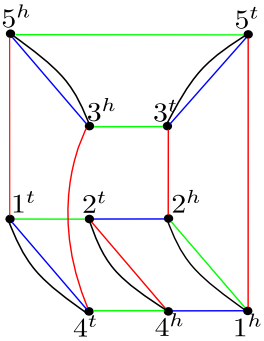
(b)

aresta e	s(e)	$\lambda(e)$
1^t5^h	1.5	0.5
3^h4^t	1.5	0.5
2^t4^h	1.0	1.0
2^h3^t	1.0	1.0
1^h5^t	1.0	1.0
Total	5.0	4.0



(c)

aresta e	s(e)	$\lambda(e)$
1^t5^h	1.5	0.5
3^h4^t	1.5	0.5
2^t4^h	1.5	0.5
1^h2^h	2.0	0.0
3^t5^t	1.5	0.5
Total	8.0	2.0



(d)

aresta e	s(e)	$\lambda(e)$
1^t4^t	2.0	0.0
3^h5^h	2.0	0.0
3^t5^t	2.0	0.0
1^h2^h	2.0	0.0
2^t4^h	2.0	0.0
Total	10.0	0.0

Figura 4.6: (a) Dado um grafo G representado pelos emparelhamentos perfeitos $\Pi_1 = \{1^h5^t, 5^h1^t, 3^h4^t, 4^h2^t, 2^h3^t\}$, $\Pi_2 = \{3^h5^h, 5^t3^t, 2^t2^h1^h4^h, 4^t1^t\}$ e $\Pi_3 = \{1^h2^h2^t1^t, 4^t4^h, 3^t3^h, 5^t5^h\}$. (b) Como não há arestas confiáveis, as arestas do emparelhamento Π_1 são escolhidas como solução inicial. As arestas 2^t4^h (c) e 3^t5^t (d) são escolhidas e a solução devolvida é $\Gamma = \{1^h2^h, 2^t4^h, 4^t1^t, 3^h5^h, 5^t3^t\}$, o qual é a solução ótima.

4.4.1 Versões da pontuação das arestas

Na tentativa de encontrar mais ciclos e, consequentente, obter soluções mais próximas das soluções ótimas, foram acrescidas restrições nas operações do algoritmo 5, visando o aperfeiçoamento da heurística. Todas as versões apresentadas a seguir contêm uma restrição em comum: se existe uma aresta $uv \in E(G)$ tal que $\mu(uv) = 2$, então a aresta u_1v_1 fará parte da solução se $u_1v_1 \in E(G)$. Essa restrição consome tempo $O(n^2)$, pois percorre as $3n$ arestas de G e verifica se $u_1v_1 \in E(G)$.

Vizinhos adjacentes

A ideia desta versão é a seguinte. Dado um grafo cúbico 3-aresta-colorido G representando o conjunto de emparelhamentos perfeitos $\mathbf{\Pi} = \{\Pi_1, \Pi_2, \Pi_3\}$, a heurística inicia fixando as arestas confiáveis $\mathcal{R}$ como parte da solução. Além disso, fixa as arestas que se encontram na seguinte condição: se uma aresta $uv \in G$ tem vizinhos que são adjacentes entre si, ou seja, $\{uu_1, vv_1\}$ e $\{uu_2, vv_2\}$ e as adjacências $u_1v_1, u_2v_2 \in E(G)$, então as arestas uv, u_1v_1 e u_2v_2 farão parte da solução. Em seguida, o algoritmo escolhe as arestas restantes aleatoriamente para fazer parte da solução inicial. Após a construção da solução inicial, o algoritmo prossegue incrementando a pontuação das arestas, enquanto o potencial de ciclo das mesmas é maior que 0.

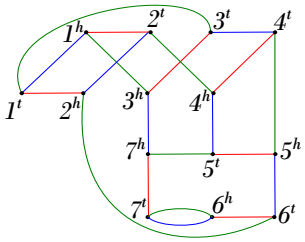
Algoritmo 6 Pontuação das arestas (vizinhos adjacentes)

Entrada: Grafo cúbico 3-aresta-colorido G obtido dos emparelhamentos perfeitos em $\mathbf{\Pi}$
Saída: Grafo 1-regular Γ sobre $V(G)$ tal que $s(\Gamma)$ é máximo considerando fixas as arestas confiáveis

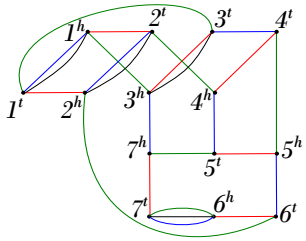
1: Seja $\mathcal{R}$ o conjunto das arestas confiáveis G
2: Seja $\mathcal{A}$ o conjunto das arestas vizinhas das arestas em $\mathcal{R}$ e adjacentes entre si
3: $\mathcal{R} \leftarrow \mathcal{R} \cup \mathcal{A}$
4: Seja Γ um emparelhamento perfeito em G formado por $\mathcal{R}$ e arestas restantes de G de forma arbitrária
5: Compute $s(uv), \lambda(uv)$ para cada aresta uv em Γ
6: **se** existe alguma aresta uv em Γ tal que $\lambda(uv) > 0$ **então**
7: Sejam uu_1, vv_1 arestas i-coloridas, $u_1v_1 \notin \Gamma$, $i \in \{1, 2, 3\}$
8: Sejam u_1u_2, v_1v_2 arestas em Γ
9: **se** $u_1u_2, v_1v_2 \notin \mathcal{R}$ **então**
10: $\Gamma = \Gamma + \{u_1v_1, u_2v_2\} - \{u_1u_2, v_1v_2\}$
11: Atualize $s(uv), \lambda(uv)$
12: Atualize s,λ para arestas irmãs de u_1u_2,v_1v_2,u_1v_1,u_2v_2
13: Compute $s(u_1v_1), \lambda(u_1v_1)$ e $s(u_2v_2), \lambda(u_2v_2)$
14: Repita as linhas 4–11 enquanto $s(\Gamma)$ pode ser incrementado sem remover as arestas em $\mathcal{R}$
15: **devolva** Γ

O algoritmo nesta versão tem tempo de execução $O(n^2)$, sendo $2n$ o número de vértices do grafo de entrada G. Na linha 2, o procedimento de fixar as arestas de acordo com a condição consome tempo $O(n^2)$, pois percorre todas as $3n$ arestas e cada verificação tem tempo $O(n)$.

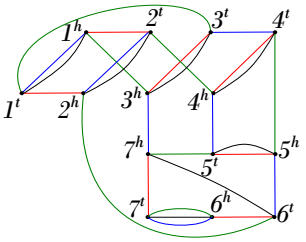
Um exemplo mostrando esta versão é apresentado na figura 4.7.



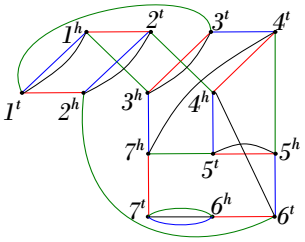
(a)



(b)



(c)



(d)

aresta e	s(e)	$\lambda(e)$
1^t1^h	2.0	0.0
2^t2^h	1.5	0.5
3^t3^h	1.5	0.5
4^t4^h	1.0	1.0
5^t5^h	1.0	1.0
6^h7^t	2.5	0.0
6^t7^h	0.5	1.0
Total	10.0	4.0

aresta e	s(e)	$\lambda(e)$
1^t1^h	2.0	0.0
2^t2^h	2.0	0.0
3^t3^h	2.0	0.0
4^t7^h	1.0	0.5
4^h6^t	1.0	0.5
5^t5^h	2.0	0.5
6^h7^t	2.0	0.5
Total	12.0	2.0

Figura 4.7: (a) Emparelhamentos perfeitos $\Pi_1 = \{1^h2^t, 2^h1^t, 3^t3^h, 4^t4^h, 5^t5^h, 6^t6^h, 7^t7^h\}$, $\Pi_2 = \{1^t1^h, 2^t2^h, 4^h5^t, 5^h6^t, 6^h7^t, 7^h3^h, 3^t4^t\}$ e $\Pi_3 = \{6^h7^t, 7^h5^t, 5^h4^t, 4^h2^t, 2^h6^t, 1^h3^h, 3^t1^t\}$. (b) As arestas vizinhas $1^t1^h, 2^t2^h$ e 3^t3^h são fixadas juntamente com a aresta confiável 6^h7^t. (c) A solução inicial é completada com as arestas $4^t4^h, 5^t5^h$ e 6^t7^h. (d) A aresta 3^t3^h é escolhida para aumentar a pontuação resultando em $\Gamma = \{1^t1^h, 2^t2^h, 3^t3^h, 4^h6^t, 6^h7^t, 7^h4^t, 5^t5^h\}$, o qual é uma solução ótima.

Sorteio das arestas

A ideia desta versão da heurística é a seguinte. Dado um grafo cúbico 3-aresta-colorido G representando o conjunto de emparelhamentos perfeitos $\mathbf{\Pi} = \{\Pi_1, \Pi_2, \Pi_3\}$, o algoritmo inicia fixando as arestas confiáveis $\mathcal{R}$ como parte da solução. Em seguida, completa o conjunto de arestas Γ sorteando as arestas restantes pertencentes a G. O sorteio é feito da seguinte forma: Seja $S = \{uv : uv \in E(G) - \Gamma\}$ o conjunto das arestas restantes. A cada embaralhamento do conjunto S, uma aresta $uv \in S$ com $u, v \notin V(\Gamma)$ é selecionada para fazer parte da solução inicial. Após o sorteio das arestas restantes e formado o conjunto de arestas da solução inicial, o algoritmo computa as pontuações e potenciais de ciclos de cada aresta $uv \in \Gamma$ e tenta aumentar a pontuação do conjunto. enquanto há arestas uv com $\lambda(uv) > 0$.

Algoritmo 7 Pontuação das arestas (Sorteio das arestas)

Entrada: Grafo cúbico 3-aresta-colorido G obtido dos emparelhamentos perfeitos em $\mathbf{\Pi}$

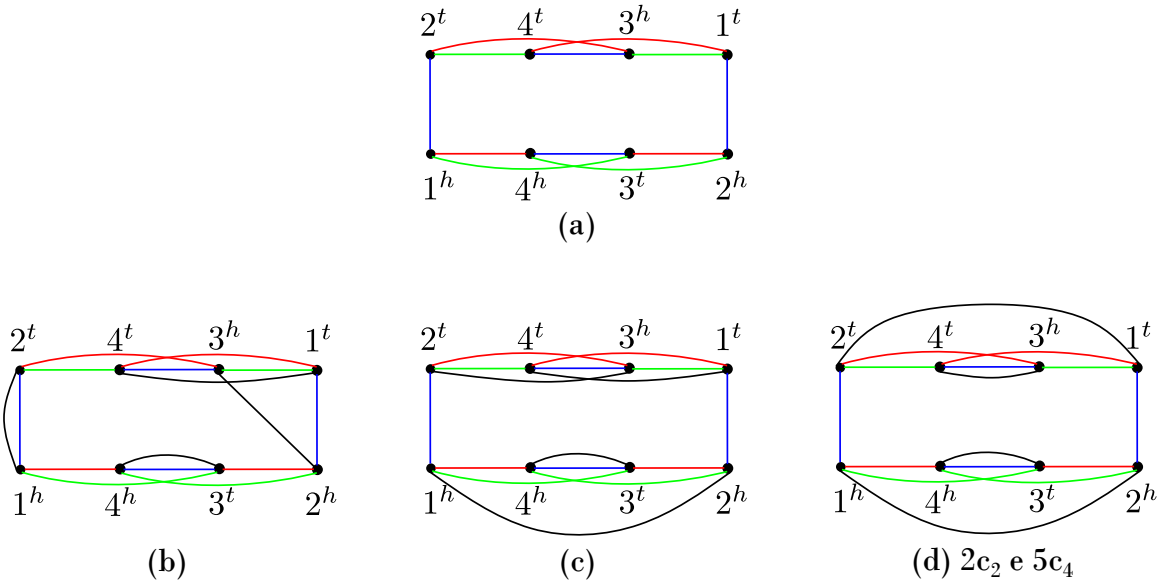
Saída: Grafo 1-regular Γ sobre $V(G)$ tal que $s(\Gamma)$ é máximo considerando fixas as arestas confiáveis

1: Seja $\mathcal{R}$ o conjunto das arestas confiáveis G
2: $\Gamma \leftarrow \mathcal{R}$
3: $S \leftarrow \{uv : uv \in E(G) - \Gamma\}$
4: Embaralhe S e selecione $uv \in S$, tal que $u, v \notin V(\Gamma)$ e faça $\Gamma \leftarrow \Gamma + \{uv\}$ até se tornar um emparelhamento perfeito em G
5: Compute $s(uv), \lambda(uv)$ para cada aresta uv em Γ
6: **se** existe alguma aresta uv em Γ tal que $\lambda(uv) > 0$ **então**
7: Sejam uu_1, vv_1 arestas i-coloridas, $u_1v_1 \notin \Gamma$, $i \in \{1, 2, 3\}$
8: Sejam u_1u_2, v_1v_2 arestas em Γ
9: **se** $u_1u_2, v_1v_2 \notin \mathcal{R}$ **então**
10: $\Gamma = \Gamma + \{u_1v_1, u_2v_2\} - \{u_1u_2, v_1v_2\}$
11: Atualize $s(uv), \lambda(uv)$
12: Atualize s,λ para arestas irmãs de u_1u_2,v_1v_2,u_1v_1,u_2v_2
13: Compute $s(u_1v_1), \lambda(u_1v_1)$ e $s(u_2v_2), \lambda(u_2v_2)$
14: Repita as linhas 4–11 enquanto $s(\Gamma)$ pode ser incrementado sem remover as arestas em $\mathcal{R}$
15: **devolva** Γ

Na linha 5, o procedimento de embaralhar e selecionar o subconjunto tem tempo de execução $O(n)$. O algoritmo nesta versão consome tempo $O(n^2)$. Veja a figura 4.8 para um exemplo.

Maior potencial de ciclo, menor pontuação

A ideia desta versão da heurística é a seguinte. Dado um grafo cúbico 3-aresta-colorido G representando o conjunto de emparelhamentos perfeitos $\mathbf{\Pi} = \{\Pi_1, \Pi_2, \Pi_3\}$, o algoritmo inicia fixando as arestas confiáveis $\mathcal{R}$ e completa o conjunto de arestas da mediana Γ com arestas restantes aleatórias, formando a solução inicial. Para aumentar a pontuação das arestas, o algoritmo escolhe a aresta $uv \in \Gamma$ tal que a aresta uv tem o menor potencial de ciclo, com $\lambda(uv) > 0$, e maior pontuação.



aresta e	s(e)	λ(e)
1^h2^t	1.0	1.0
2^h3^h	0.5	1.0
3^t4^h	1.0	1.0
1^t4^t	1.5	0.5
Total	4.0	3.5

aresta e	s(e)	λ(e)
1^h2^h	1.0	0.5
1^t4^t	1.5	0.5
3^t4^h	2.0	0.0
2^t3^h	1.5	0.5
Total	6.0	1.5

aresta e	s(e)	λ(e)
1^t2^t	1.5	0.0
3^h4^t	2.0	0.0
3^t4^h	2.0	0.0
1^h2^h	1.5	0.0
Total	7.0	0.0

Figura 4.8: (a) Emparelhamentos perfeitos $\Pi_1 = \{1^h4^h, 4^t1^t, 3^h2^t, 2^h3^t\}$, $\Pi_2 = \{1^h2^t, 2^h1^t, 3^h4^t, 4^h3^t\}$ e $\Pi_3 = \{1^h3^t, 3^h1^t, 2^h4^h, 4^t2^t\}$. (b) As arestas sorteadas 1^h2^t, 2^h3^h, 3^t4^h e 4^t1^t formam a solução inicial. As arestas 3^t4^h (c) e 1^h2^h (d) são escolhidas para aumentar a pontuação resultando na mediana $\Gamma = \{1^h2^h, 2^t1^t, 3^h4^t, 4^h3^t\}$, que é uma solução ótima.

Algoritmo 8 Pontuação das arestas (Maior potencial de ciclo, menor pontuação)

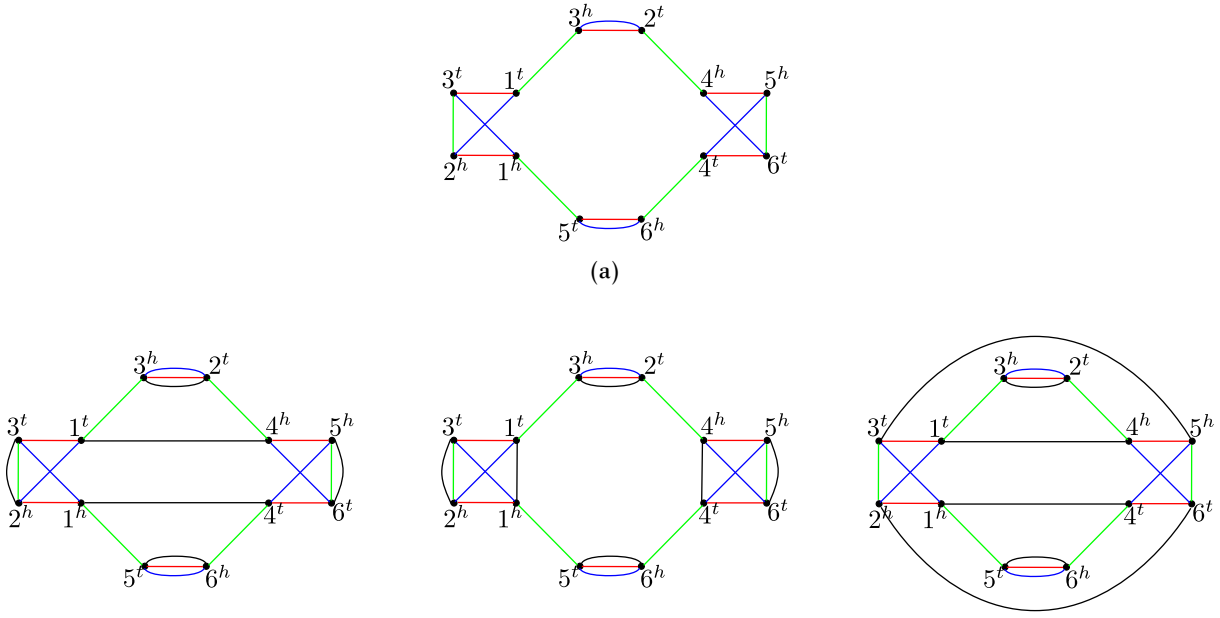
Entrada: Grafo cúbico 3-aresta-colorido G obtido dos emparelhamentos perfeitos em $\mathbf{\Pi}$

Saída: Grafo 1-regular Γ sobre $V(G)$ tal que $s(\Gamma)$ é máximo considerando fixas as arestas confiáveis

1: Seja $\mathcal{R}$ o conjunto das arestas confiáveis G
2: Seja Γ um emparelhamento perfeito em G formado por $\mathcal{R}$ e arestas restantes de G de forma arbitrária
3: Compute $s(uv), \lambda(uv)$ para cada aresta uv em Γ
4: **se** existe alguma aresta uv em Γ tal que $\lambda(uv) > 0$ **então**
5: Escolha uma aresta uv que tem o menor potencial de ciclo e maior pontuação
6: Sejam uu_1, vv_1 arestas i-coloridas, $u_1v_1 \notin \Gamma$, $i \in \{1, 2, 3\}$
7: Sejam u_1u_2, v_1v_2 arestas em Γ
8: **se** $u_1u_2, v_1v_2 \notin \mathcal{R}$ **então**
9: $\Gamma = \Gamma + \{u_1v_1, u_2v_2\} - \{u_1u_2, v_1v_2\}$
10: Atualize $s(uv), \lambda(uv)$
11: Atualize s,λ para arestas irmãs de u_1u_2,v_1v_2,u_1v_1,u_2v_2
12: Compute $s(u_1v_1), \lambda(u_1v_1)$ e $s(u_2v_2), \lambda(u_2v_2)$
13: Repita as linhas 4–11 enquanto $s(\Gamma)$ pode ser incrementado sem remover as arestas em $\mathcal{R}$
14: **devolva** Γ

Na linha 5, o procedimento para selecionar a aresta que tem o maior potencial de ciclo e a menor pontuação tem tempo $O(n)$. A inclusão dessa restrição não altera o tempo do algoritmo $O(n^2)$.

Um exemplo mostrando esta versão é apresentado na figura 4.9.



aresta e	s(e)	λ(e)
2^t3^h	2.5	0.0
1^t4^h	0.5	1.0
1^h4^t	0.5	1.0
5^t6^h	2.5	0.0
5^h6^t	1.0	1.0
2^h3^t	1.0	1.0
Total	8.0	4.0

aresta e	s(e)	λ(e)
2^t3^h	2.0	0.5
5^t6^h	2.0	0.5
1^t1^h	1.0	0.5
2^h3^t	2.0	0.0
4^h4^t	1.0	0.5
5^h6^t	2.0	0.0
Total	10.0	2.0

aresta e	s(e)	λ(e)
2^t3^h	2.5	0.0
5^t6^h	2.5	0.0
1^t4^h	1.5	0.0
2^h6^t	1.5	0.0
1^h4^t	1.5	0.0
3^t5^h	1.5	0.0
Total	11.0	0.0

Figura 4.9: (a) Emparelhamentos perfeitos $\Pi_1 = \{3^h2^t, 2^h1^h, 1^t3^t, 4^h5^h, 5^t6^h, 6^t4^t\}$, $\Pi_2 = \{1^h3^t, 3^h2^t, 2^h1^t, 4^h6^t, 6^h5^t, 5^h4^t\}$ e $\Pi_3 = \{1^h5^t, 5^h6^t, 6^h4^t, 4^h2^t, 2^h3^t, 3^h1^t\}$. (b) Solução inicial $\Gamma = \{1^h4^t, 4^h1^t, 5^h6^t, 6^h5^t, 2^h3^t, 3^h2^t\}$. (c) Escolhendo 5^h6^t ou 2^h3^t resulta na pontuação $s(\Gamma) = 10$. (d) Escolhendo a aresta 1^t4^h ou 1^h4^t resulta na pontuação $s(\Gamma) = 11$, e a solução é $\Gamma = \{(1\ 4), (3\ 2\ 6\ 5)\}$, que é uma solução ótima.

Arestas 4-ciclos

Nesta versão, o algoritmo tem a seguinte ideia. Dado um grafo cúbico 3-aresta-colorido G representando o conjunto de emparelhamentos perfeitos $\mathbf{\Pi} = \{\Pi_1, \Pi_2, \Pi_3\}$, a heurística, além de fixar as arestas confiáveis $\mathcal{R}$, fixa arestas que formam 4-ciclos, ou seja, se $uv \in E(G)$ (com as suas adjacências uu_1 e vv_1) e $u_1v_1 \in E(G)$, então as arestas uv e u_1v_1 farão parte da solução juntamente com as arestas confiáveis $\mathcal{R}$.

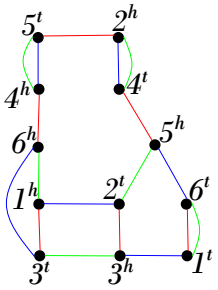
Algoritmo 9 Pontuação das arestas (Arestas 4-ciclos)

Entrada: Grafo cúbico 3-aresta-colorido G obtido dos emparelhamentos perfeitos em $\mathbf{\Pi}$

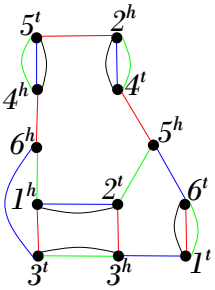
Saída: Grafo 1-regular Γ sobre $V(G)$ tal que $s(\Gamma)$ é máximo considerando fixas as arestas confiáveis e arestas que conformam 4-ciclos

1: Seja $\mathcal{R}$ o conjunto das arestas confiáveis G
2: Seja $\mathcal{C}$ o conjunto das arestas que formam 4-ciclos
3: $\mathcal{R} \leftarrow \mathcal{R} \cup \mathcal{C}$
4: Seja Γ um emparelhamento perfeito em G formado por $\mathcal{R}$ e arestas restantes de G de forma arbitrária
5: Compute $s(uv), \lambda(uv)$ para cada aresta uv em Γ
6: **se** existe alguma aresta uv em Γ tal que $\lambda(uv) > 0$ **então**
7: Sejam uu_1, vv_1 arestas i-coloridas, $u_1v_1 \notin \Gamma$, $i \in \{1, 2, 3\}$
8: Sejam u_1u_2, v_1v_2 arestas em Γ
9: **se** $u_1u_2, v_1v_2 \notin \mathcal{R}$ **então**
10: $\Gamma = \Gamma + \{u_1v_1, u_2v_2\} - \{u_1u_2, v_1v_2\}$
11: Atualize $s(uv), \lambda(uv)$
12: Atualize s,λ para arestas irmãs de u_1u_2,v_1v_2,u_1v_1,u_2v_2
13: Compute $s(u_1v_1), \lambda(u_1v_1)$ e $s(u_2v_2), \lambda(u_2v_2)$
14: Repita as linhas 4–11 enquanto $s(\Gamma)$ pode ser incrementado sem remover as arestas em $\mathcal{R}$
15: **devolva** Γ

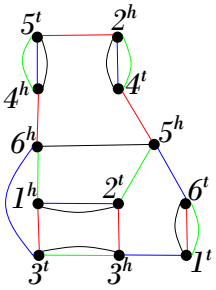
Na linha 2, o procedimento de escolher as arestas que formam 4-ciclos tem tempo $O(n^2)$, pois o consumo de tempo para percorrer as arestas de G e verificar a existência de 4-ciclos é $O(n)$. Portanto, o tempo de execução do algoritmo nesta versão é $O(n^2)$. Veja um exemplo do algoritmo 9 na figura 4.10.



(a)



(b)



(c)

aresta e	s(e)	$\lambda(e)$
1^h2^t	2.0	0.0
3^t3^h	1.5	0.5
1^t6^t	2.0	0.5
5^h6^h	0.5	1.0
4^h5^t	2.0	0.5
2^h4^t	2.0	0.5
Total	10.0	3.0

Figura 4.10: (a) Emparelhamentos perfeitos $\Pi_1 = \{1^h3^t, 3^h2^t, 2^h5^t, 5^h4^t, 4^h6^h, 6^t1^t\}$, $\Pi_2 = \{2^h4^t, 4^h5^t, 5^h6^t, 6^h3^t, 3^h1^t, 1^h2^t\}$ e $\Pi_3 = \{3^t3^h, 2^h4^t, 4^h5^t, 5^h2^t, 1^h6^h, 6^t1^t\}$. (b) Além de fixar as arestas confiáveis $4^h5^t, 2^h4^t$ e 1^t6^t, também são fixadas as arestas que formam 4-ciclos: $\{1^h2^t, 3^t3^h\}$. (c) A solução inicial é completada com a adição da aresta 5^h6^h e, consequentemente, a mediana é $\Gamma = \{1^h2^t, 2^h4^t, 4^h5^t, 5^h6^h, 6^t1^t, 3^t3^h\}$, que é uma solução ótima.

Capítulo 5

Experimentos

Após ter apresentado os algortitmos exatos e heurísticas combinatórias, realizamos experimentos com os algoritmos propostos, fizemos uma análise do desempenho e verificamos alguns pontos que poderão ser ajustados visando melhorias nas soluções das instâncias do problema.

Simulamos múltiplos genomas a fim de (i) traçar os limites onde os nossos PLIs (algoritmos 1 e 2) podem funcionar em tempo razoável enquanto fornece uma precisão aceitável, e (ii) avaliar a qualidade e o tempo de execução das heurísticas (algoritmos 3, 4 e 5) incluindo as versões da pontuação das arestas (algoritmos 6, 7, 8 e 9). Os experimentos foram realizados utilizando a memória da CPU com frequência de 3.60GHz. Implementamos as heurísticas em Python 3 e foi usado o Gurobi 9.0.2 como *solver* da PLI com 8 *cores* e o tempo limite de 1 hora.

5.1 Programação linear inteira

Dado o genoma raiz com n marcadores, o **trio descendente** é o conjunto de três genomas, cada um gerado pela simulação de $\frac{n}{100} \times k$ operações DCJ aleatórios independentes no genoma raiz, sendo k a porcentagem do tamanho n do genoma raiz.

Primeiro, geramos genomas raiz com $50, 100, 150, \ldots, 500$ marcadores distribuídos em vários cromossomos circulares e, em seguida, para cada genoma raiz, quatro trios descendentes variando k em $\{10,\ 15,\ 20,\ 25\}$. Para este conjunto de dados, o algoritmo 2 encontra soluções mais rapidamente que o algoritmo 1 (veja a Figura 5.1).

Ao utilizar o *solver*, no comando de execução, além de especificar o parâmetro de tempo limite, escolhemos o método concorrente não determinístico ($method = 3$). O método utiliza solucionadores em vários encadeamentos simultaneamente e escolhem aquele que termina primeiro. A inclusão deste método faz o *solver* ser rápido [12], pois é um otimizador simultâneo. O limite de tempo parece depender mais do número de DCJs usados para gerar os trios que do tamanho de genomas, pois o *solver* gasta o tempo de 1 hora para o algoritmo 1 com o valor de $k = 25$. No entanto, para o algoritmo 2, o *solver* não excede o tempo limite ao encontrar soluções para este conjunto de dados.

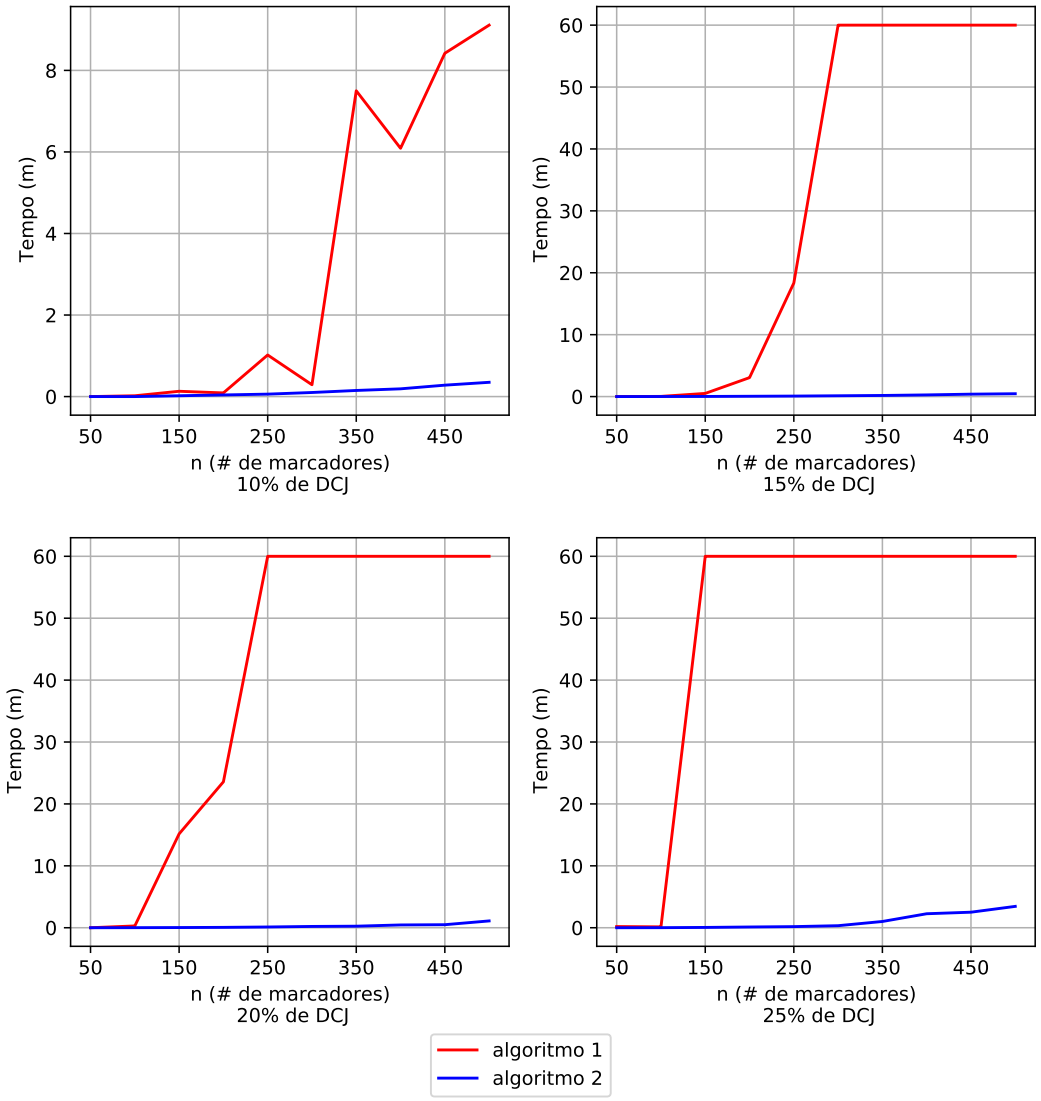


Figura 5.1: Tempo de execução do *solver* da PLI em minutos para múltiplos tamanhos de genomas e k valores.

Com o objetivo de avaliar o desempenho do algoritmo 2 para grandes genomas, geramos genomas raiz com $1000, 1100, 1200, \ldots, 2000$ marcadores distribuídos em vários cromossomos circulares e, em seguida, para cada genoma raiz, quatro trios descendentes variando k em $\{10, 15, 20, 25\}$. No comando de execução do *solver*, modificamos o tempo limite de 1 hora para 10 horas, mas mantivemos o parâmetro método com o valor 3 ($method = 3$). Para este conjunto de dados, o *solver* excede o tempo limite para genomas com $n > 1100$ marcadores quando $k = 25$ (veja a Figura 5.2).

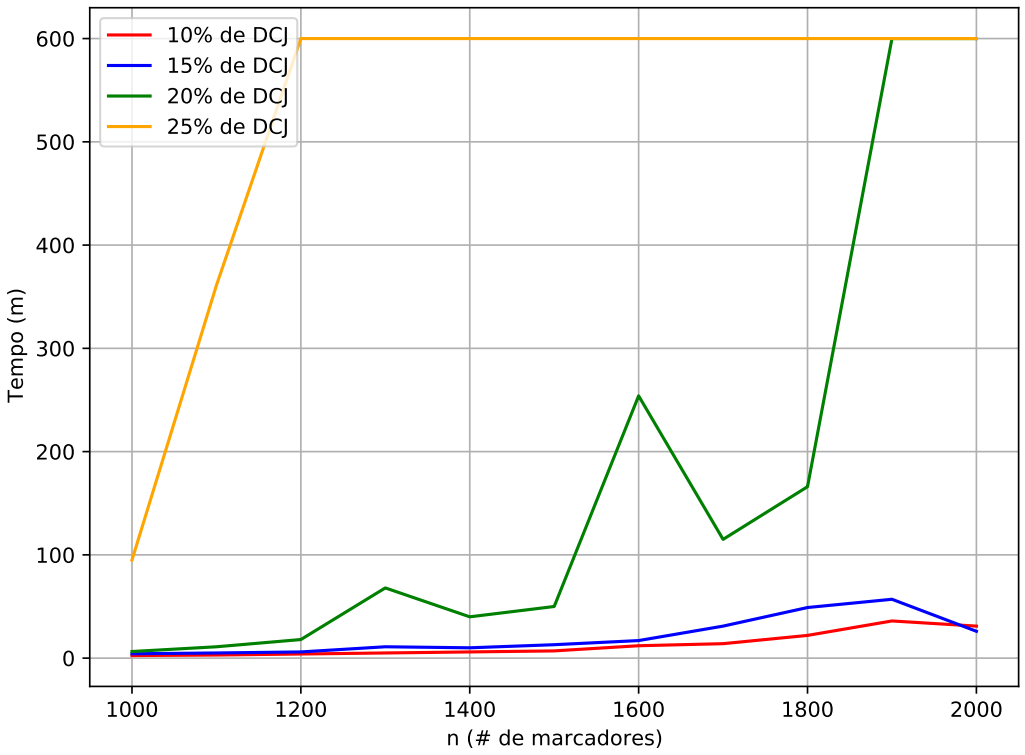


Figura 5.2: Tempo de execução do *solver* para múltiplos genomas com $1000 \leq n \leq 2000$ marcadores e k valores.

5.2 Heurísticas

Em segundo lugar, a fim de estressar as heurísticas e avaliar o seu desempenho, simulamos conjuntos de dados com grandes genomas, de 1.000 a 10.000 marcadores e $k = 25$. A Figura 5.3 mostra o tempo de execução e as distâncias-c_4 para os algoritmos Ciclos bicoloridos, Encurtamento de adjacências e Pontuação de arestas. A heurística Pontuação das arestas sempre devolve as menores distâncias-c_4, seguido do algoritmo Encurtamento de adjacências com a diferença de 0.71%, em média.

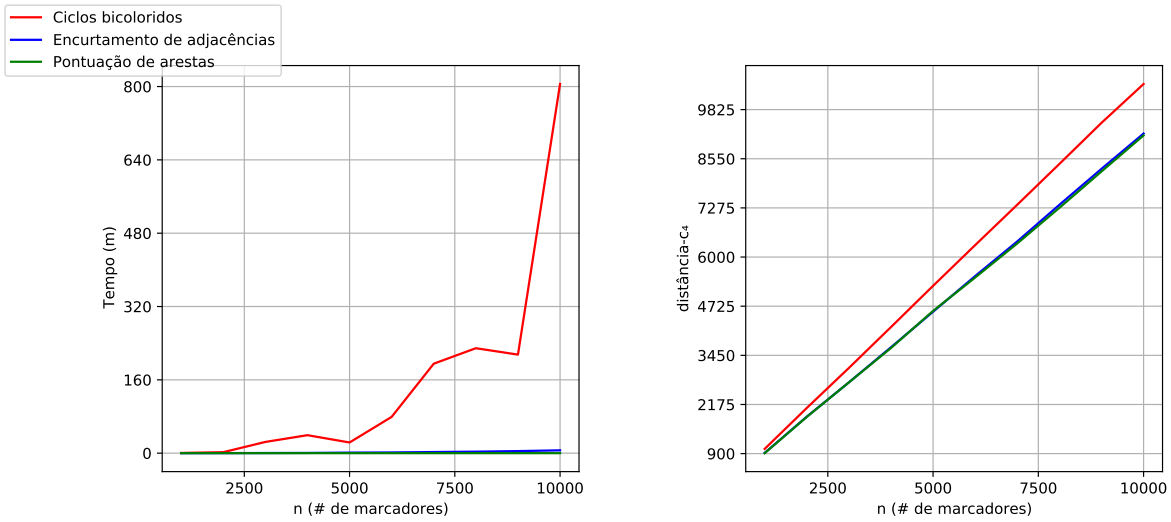


Figura 5.3: Para múltiplos genomas e $k = 25$, (a) tempo da execução das heurísticas em minutos e (b) distâncias-c_4 dadas pelas heurísticas.

Avaliamos também as versões da heurística pontuação das arestas, a quais são os Vizinhos adjacentes, maior potencial de ciclo e menor pontuação, sorteio das arestas e arestas 4-ciclos. A Figura 5.4 mostra o tempo de execução e as distâncias-c_4. Para o mesmo conjunto de dados utilizados no experimento acima, o algoritmo Vizinhos adjacentes sempre devolve as menores distâncias-c_4, seguido do algoritmo Maior potencial de ciclo e menor pontuação com a diferença de 0.2%, em média.

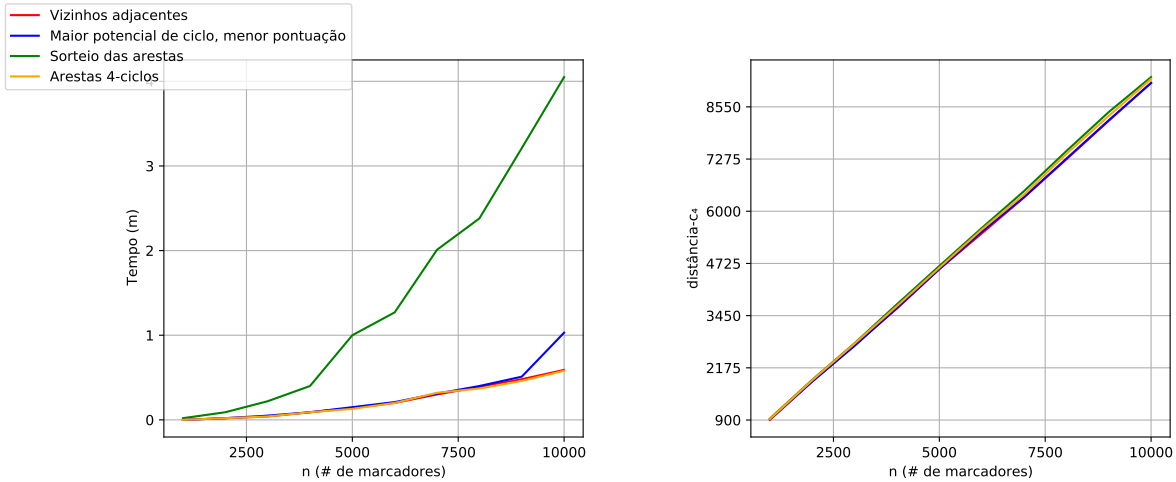


Figura 5.4: Avaliação de (a) tempo de execução e (b) distâncias-c_4 dos algoritmos Vizinhos adjacentes, Maior potencial de ciclo e menor pontuação, sorteio das arestas e arestas 4-ciclos para múltíplos tamanhos de genomas e $k = 25$.

5.2.1 Combinação das heurísticas

Realizamos experimentos com a combinação das três heurísticas (Ciclos bicoloridos, Encurtamento de adjacências e Pontuação de arestas) no mesmo conjunto de dados. Primeiramente, juntamos o Ciclos bicoloridos com o Pontuação de arestas e depois Encurtamento de adjacências com Pontuação de arestas. O resultado do primeiro algoritmo é usado como entrada de solução inicial no segundo algoritmo. A Figura 5.5 mostra a comparação dos resultados entre essas duas combinações. A combinação do Encurtamento de adjacências com Pontuação de arestas termina mais rapidamente e devolve menores distâncias-c_4 com a diferença de 3.74% em média.

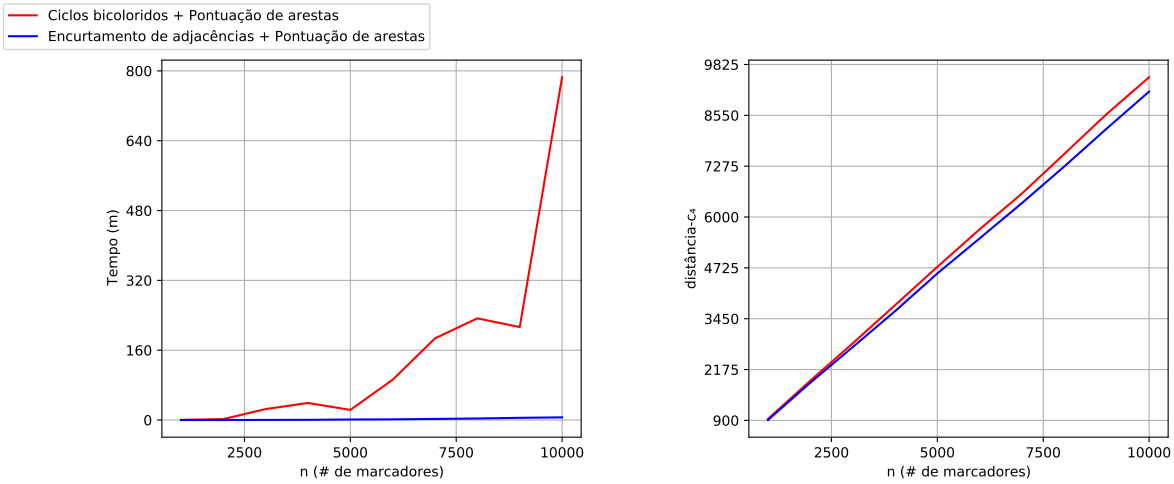


Figura 5.5: Comparação de (a) tempo de execução e (b) distância-c_4 das combinações das heurísticas: Ciclos bicoloridos, Encurtamento de adjacências e Pontuação de arestas.

Ao compararmos os resultados entre a combinação dos algoritmos Encurtamento de adjacências e Pontuação de arestas com os algoritmos Pontuação de arestas e Vizinhos adjacentes, vemos que o algoritmo Vizinhos adjacentes devolve as menores distâncias-c_4 seguido do algoritmo Pontuação das arestas com a mesma diferença de 0.3% em média (veja a Figura 5.6).

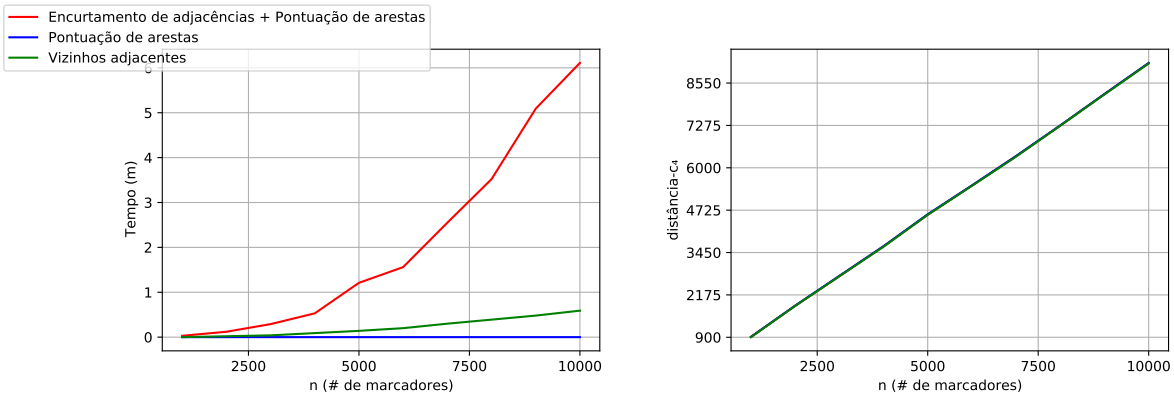


Figura 5.6: Comparação de (a) tempo de execução e (b) distância-c_4 dos três principais algoritmos.

5.3 Heurísticas e PLI

Após realizar os experimentos com as heurísticas e algoritmos exatos, comparamos as distâncias-c_4 devolvidas pelas heurísticas com as distâncias-c_4 ótimas obtidas pelo PLI. A comparação foi feita com o mesmo conjunto de dados utilizados na primeira parte dos experimentos, sendo o número de marcadores é de 50 a 500 com $k = 25$.

Primeiramente, realizamos as comparações dos resultados obtidos pelas três primeiras heurísticas (Ciclos bicoloridos, Encurtamento de adjacências e Pontuação das arestas) com as distâncias-c_4 ótimas. A Figura 5.7 mostra que a heurística Pontuação das arestas devolve distâncias-c_4 mais próximas das ótimas com a diferença de 2.8% em média. Em seguida, as heurísticas Encurtamento de adjacências e Ciclos bicoloridos têm, respectivamente, as diferenças de 5.5% e 15.9% em média em relação às distâncias-c_4 ótimas.

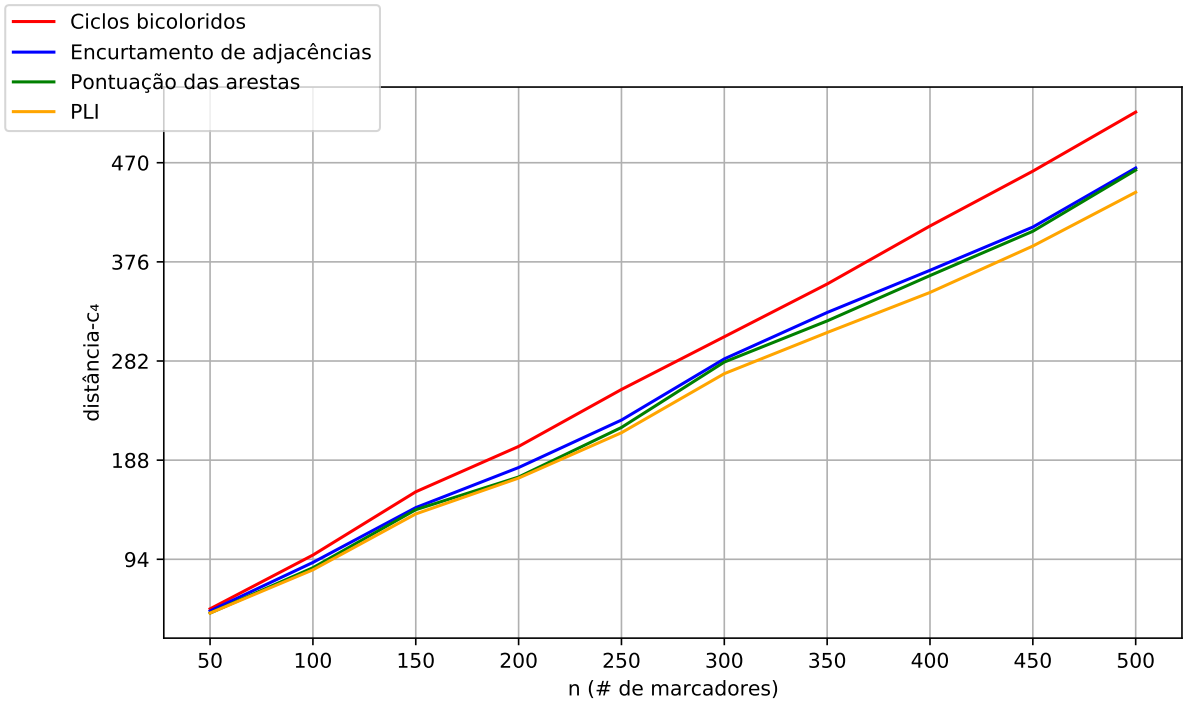


Figura 5.7: Distâncias-c_4 obtidas pelas heurísticas Ciclos bicoloridos, Encurtamento de adjacências, Pontuação das arestas e pelo PLI, com $k = 25$.

Em seguida, foi realizado a comparação das distâncias-c_4 obtidas pelas versões da Pontuação das arestas (Vizinhos adjacentes, Sorteio de arestas, Maior potencial de ciclo e menor pontuação e Arestas 4-ciclos) com as distâncias-c_4 ótimas obtidas pelo PLI. A Figura 5.8 mostra que a heurística Vizinhos adjacentes devolve distâncias-c_4 mais próximas das ótimas com média de 2.5% de diferença, seguida das heurísticas Maior potencial de ciclo e menor pontuação, Sorteio de arestas e Arestas 4-ciclos com as diferenças, respectivamente, de 3%, 3.4% e 4% em média.

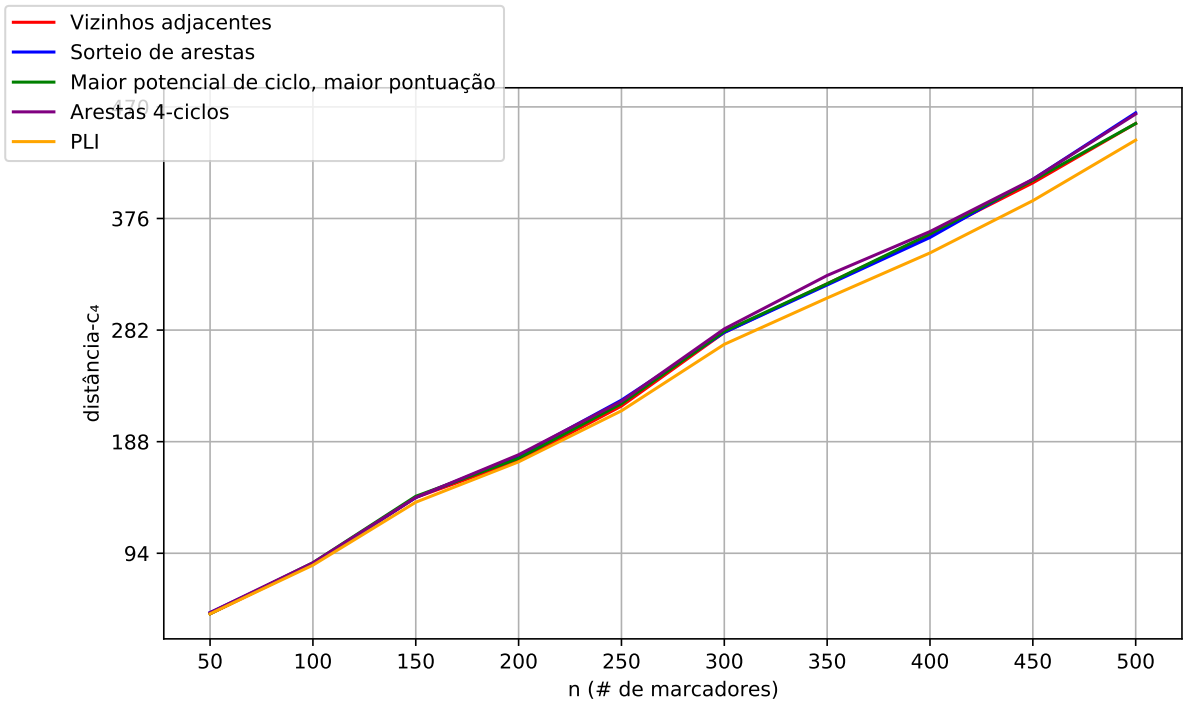


Figura 5.8: Distâncias-c_4 obtidas pelas heurísticas Vizinhos adjacentes, Sorteio de arestas, Maior potencial de ciclo e menor pontuação, Arestas 4-ciclos e pelo PLI, com $k = 25$.

Por fim, realizamos a comparação das distâncias-c_4 obtidas pelas combinações entre heurísticas Ciclos bicoloridos, Pontuação das arestas e Encurtamento de adjacências com o algoritmo PLI. A Figura 5.9 mostra que a combinação Encurtamento de adjacências e Pontuação das arestas devolvem distâncias-c_4 mais próximas das ótimas, a diferença é de 3.6% em média. A combinação das heurísticas Ciclos bicoloridos e Pontuação das arestas devolvem distâncias-c_4 com a diferença de 10.2% em média em relação às ótimas.

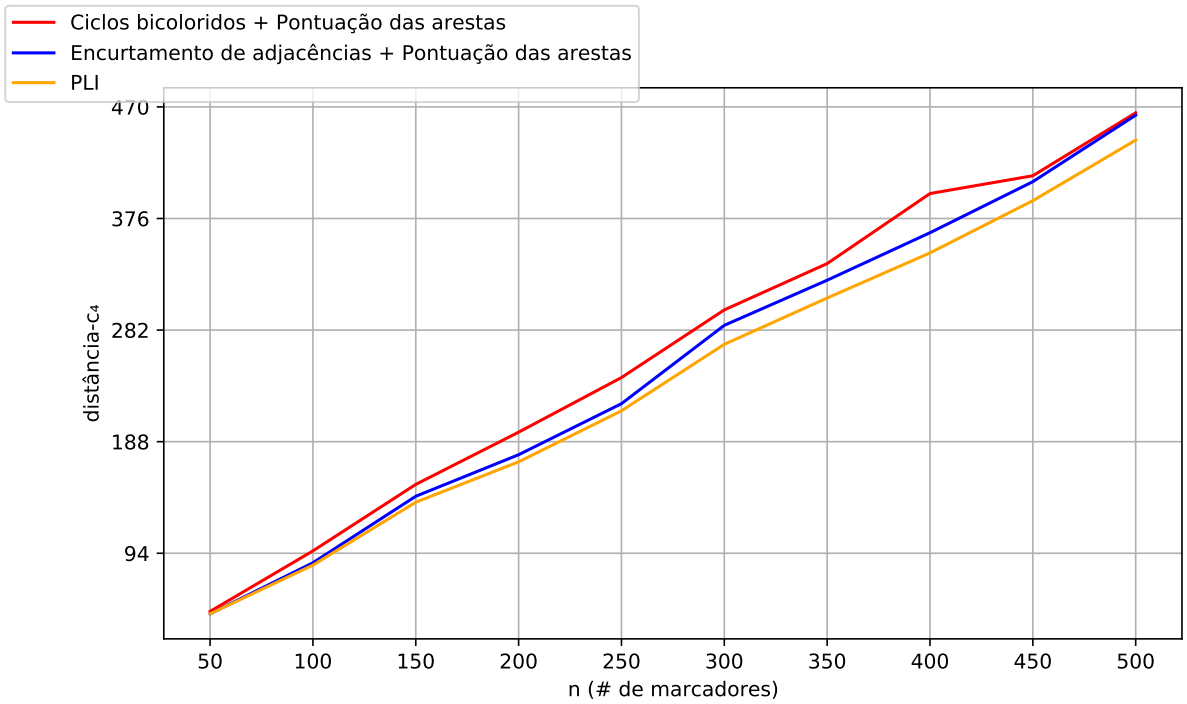


Figura 5.9: Distâncias-c_4 obtidas pelas combinações das heurísticas entre Ciclos bicoloridos, Encurtamento de adjacências e Pontuação das arestas e pelo PLI, com $k = 25$.

5.4 Considerações sobre os experimentos

A segunda versão em PLI encontra soluções ótimas para o problema da Mediana-c_4 mais rapidamente que a primeira versão. Uma explicação sobre o *solver* exceder o tempo limite e, consequentemente, não encontrar a solução ótima está no fato de que, quando o trio descendente é gerado com $k \geq 25$ operações DCJs, há uma mudança de fase, ou seja, assumir parcimônia faz com que o valor da real distância de cada genoma descendente para o genoma raiz seja maior que o número de operações DCJs. Isso faz com que a probabilidade de encontrar a mediana DCJ seja muito baixa [9].

Outro ponto importante a observar sobre os algoritmos exatos PLI é que a não especificação do método nos parâmetros do *solver* para o conjunto de dados com $n <= 400$ marcadores, o *solver* atribui $method = 1$, o qual utiliza o algoritmo *primal simplex*, encontrando soluções em tempo superior em comparação com o $method = 3$. Houve uma tentativa de realizar experimentos com dados simulados para $k = \{10, 15\}$ em que $n \geq 3000$ marcadores para verificar o desempenho do tempo, porém não foi possível devido ao tamanho da memória exigida da máquina, o *solver* não prosseguia a execução.

Sobre a heurística Pontuação das arestas, percebemos que o critério de escolher a aresta uv com $\lambda(uv) > 0$ para realizar as operações é um fator importante para encontrar soluções ótimas. Algumas versões da heurística visavam a fixação de arestas para que a pontuação (número de 2- e 4-ciclos) fosse maior do que a heurística fornece. Vemos que houve uma melhoria de 0.3% em média com a versão Vizinhos adjacentes. Acreditamos que fixar uma certa quantidade de arestas ótimas com $\mu(uv) = 1$ faz com que a heurística devolve mais 2- e 4-ciclos.

Embora as heurísticas Ciclos bicoloridos e Encurtamento de adjacências devolvam menor quantidade de 2- e 4-ciclos em relação à heurística Pontuação das arestas, acreditamos que é possível utilizar suas estratégias para descobrir arestas ótimas uv com $\mu(uv) = 1$, pois para a heurística Ciclos bicoloridos, se uma aresta uv está presente na solução dos três pares de genomas distintos, então é provável que ela seja uma aresta ótima. E para a heurística Encurtamento de adjacências, se a aresta uv satisfazer o último critério, o qual diz que o maior ciclo que ela está contido for menor ou igual ao maior ciclo que outra aresta está contido, então é possível que ela seja uma aresta ótima.

Capítulo 6

Considerações finais

O problema da mediana para genomas multicromossomais circulares é NP-difícil para operação de rearranjo DCJ [13], mas polinomial para a operação SCJ [5]. O estudo sobre o problema da mediana-c_4 visa abordar uma nova medida de rearranjo restrita chamada distância-c_4 que, dado um grafo de *breakpoint* representando o conjunto de três genomas, encontra o maior número de 2- e 4-ciclos como solução.

No estudo do problema da mediana-c_4, descobrimos limitantes inferiores e superiores para a maximização de 2- e 4-ciclos em um grafo cúbico com $2n$ vértices, sendo n o número de marcadores. Os limitantes desse problema são importantes para selecionar arestas ótimas. Com eles, afirmamos que K_2^3, G_4^3, K_2^2 e toda escada linear e circular são *decomposers* fortes, ou seja, são subgrafos que fazem parte da solução ótima de todas as instâncias do problema.

Sobre os algoritmos exatos em PLI para o problema da mediana-c_4, a especificação dos parâmetros do comando no *solver* influenciam na execução, principalmente o parâmetro *method*. O método selecionado pelo *solver* é o $method = 1$ quando o modelo PLI não é grande, o qual utiliza somente o *primal simplex*. Entretanto, quando é especificado o $method = 3$, são utilizados os métodos *barrier, primal e dual simplex*.

Do ponto de vista teórico, a complexidade computacional da mediana-c_4 ainda está em aberto, embora conjecturamos que seja NP-difícil. Os algoritmos apresentados neste trabalho dão uma perspectiva prática da mediana-c_4. Vemos que o algoritmo 9 utiliza o conceito de subgrafos adequados [15] ao fixar arestas que formam 4-ciclos. Entretanto, nem toda aresta selecionada que forma um 4-ciclo resulta na solução ótima. É interessante aprofundar as relações entre esse problema e a mediana DCJ [15], e encontrar *decomposers* e subgrafos adequados restritos ao problema da mediana-c_4.

Além disso, é interessante descobrir em qual situação o conjunto de três genomas representam um grafo cúbico 3-aresta-colorido conexo, pois os dados simulados gerados, de acordo com o número de DCJs, resultaram em grafos que continham mais de um componente. Acreditamos que o grafo conexo influencia positivamente no desempenho das heurísticas apresentadas, pois há situações em que o grafo contém mais de um componente e uma aresta de solução ótima está ligando dois componentes, e as heurísticas dificilmente vão selecionar essa aresta como candidata à solução final.

Uma análise de dados reais, de organismos mais complexos, pode fornecer mais informações sobre o comportamento desta medida na prática. É possível estender este trabalho a ge-

nomas multicromossomais lineares, utilizando uma abordagem para lidar com equivalência, o *capping* [14, 7, 10], na qual podemos transformar, de forma segura, os cromossomos lineares em circulares preservando as distâncias.

Lista de Figuras

Lista de Algoritmos

Referências Bibliográficas

[1] V. Bafna and P. A. Pevzner. Genome rearrangements and sorting by reversals. *SIAM J Comput*, 25(2):272–289, 1996.

[2] A. Bergeron, J. Mixtacki, and J. Stoye. A unifying view of genome rearrangements. In *Proc. of WABI*, volume 4175 of *LNBI*, pages 163–173, 2006.

[3] A. Caprara. The reversal median problem. *INFORMS J Comput*, 15:93—113, 2003.

[4] D. Doerr, M. Balaban, P. Feijão, and C. Chauve. The gene family-free median of three. *Algorithm Mol Biol*, 12(1):1–14, 2017.

[5] P. Feijão and J. Meidanis. SCJ: A breakpoint-like distance that simplifies several rearrangement problems. *IEEE/ACM Trans Comput Biol Bioinf*, 8(5):1318–1329, 2011.

[6] G. Fertin, A. Labarre, I. Rusu, E. Tannier, and S. Vialette. *Combinatorics of Genomes Rearrangements*. The MIT Press, 2009.

[7] S. Hannenhalli and P. Pevzner. Transforming men into mice (polynomial algorithm for genomic distance problem). In *Proc. of FOCS 1995*, pages 581–592, 1995.

[8] R. C. Hardison. Comparative genomics. *PLOS Biology*, 1(58):156–160, november 2003.

[9] A. Jamshidpey and D. Sankoff. Phase change for the accuracy of the median value in estimating divergence time. In *Proc. of BMC*, LNBI, 2013.

[10] G. Jean and M. Nikolski. Genome rearrangements: a correct algorithm for optimal capping. *Inform Process Letters*, 104(1):14–20, 2007.

[11] J. Kováč. On the complexity of rearrangement problems under the breakpoint distance. *J Comput Biol*, 21(1):1–15, 2013.

[12] LLC, Gurobi Optimization. Method. https://www.gurobi.com/documentation/9.5/refman/method.html, 2020.

[13] E. Tannier, C. Zheng, and D. Sankoff. Multi-chromosomal median and halving problems under different genomic distances. *BMC Bioinformatics*, 10(120), 2009.

[14] A. W. Xu. DCJ median problems on linear multichromosomal genomes: Graph representation and fast exact solutions. In *Proc. of RECOMB-CG*, volume 5817 of *LNCS*, pages 70–83, 2009.

[15] A. W. Xu. A fast and exact algorithm for the median of three problem: A graph decomposition approach. *J Comput Biol*, 16(10):1369–1381, 2009.

[16] S. Yancopoulos, O. Attie, and R. Friedberg. Efficient sorting of genomic permutations by translocation, inversion and block interchanges. *Bioinformatics*, 21(16):3340–3346, 2005.