Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Um algoritmo algébrico para o Problema da Distância de Transposição em Rearranjo de Genomas

Luiz Augusto Garcia da Silva

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Orientadora
Prof.ª Dr.ª Maria Emília M. T. Walter

Brasília
2013

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Mestrado em Informática

Coordenador: Prof. Dr. Ricardo Pezzuol Jacobi

Banca examinadora composta por:

Prof.ª Dr.ª Maria Emília M. T. Walter (Orientadora) — CIC/UnB

Prof. Dr. Norai Romeu Rocco — MAT/UnB

Prof. Dr. Zanoni Dias — IC/Unicamp

CIP — Catalogação Internacional na Publicação

Silva, Luiz Augusto Garcia da.

Um algoritmo algébrico para o Problema da Distância de Transposição em Rearranjo de Genomas / Luiz Augusto Garcia da Silva. Brasília : UnB, 2013.
133 p. : il. ; 29,5 cm.

Dissertação (Mestrado) — Universidade de Brasília, Brasília, 2013.

1. Rearranjo de Genomas, 2. Problema da Distância de Transposição, 3. Álgebra, 4. Grupos de Permutações, 5. Algoritmos de aproximação;

CDU 004

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro — Asa Norte
CEP 70910-900
Brasília–DF — Brasil

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Um algoritmo algébrico para o Problema da Distância de Transposição em Rearranjo de Genomas

Luiz Augusto Garcia da Silva

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Prof.ª Dr.ª Maria Emília M. T. Walter (Orientadora)
CIC/UnB

Prof. Dr. Norai Romeu Rocco
MAT/UnB

Prof. Dr. Zanoni Dias
IC/Unicamp

Prof. Dr. Ricardo Pezzuol Jacobi
Coordenador do Mestrado em Informática

Brasília, 01 de dezembro de 2013

Dedicatória

A meus pais, Paulo e Maria, que nunca mediram esforços para que seus filhos tivessem uma boa educação.

Agradecimentos

À minha família, em especial à minha esposa, Juliana Capella, pelo apoio, incentivo e paciência durante todo o desenvolvimento deste trabalho.

À minha orientadora, Professora Maria Emília, pela oportunidade, pela confiança e sobretudo pela disposição em me ajudar, mesmo nos momentos mais difíceis.

Ao Professor Noraí Rocco, pelos conselhos e pelas inúmeras lições particulares de Álgebra que eu tive o privilégio de ter.

Resumo

Em Biologia Computacional, eventos mutacionais afetando grandes porções de um genoma são estudados na área de Rearranjo de Genomas. Particularmente, a *transposição* é um evento mutacional que troca de posição dois blocos contíguos de genes em um cromossomo. Este evento gera o *problema da distância de transposição* (PDT), que consiste em encontrar o número mínimo de transposições necessárias para transformar um cromossomo em outro. Recentemente, foi mostrado que o PDT é $\mathcal{NP}$-difícil.

Na literatura, muitos algoritmos foram propostos para resolver este problema, seguindo abordagens diferentes. Neste trabalho, utilizaremos o formalismo algébrico proposto por Meidanis e Dias, para a modelagem de cromossomos e transposições, e resultados clássicos de Grupos de Permutações para propor um algoritmo de aproximação com razão 2 para o problema da distância de transposição.

Embora existam algoritmos com razão de aproximação melhores, a contribuição do presente trabalho é teórica, pois propõe uma solução para o problema da distância de transposição utilizando apenas resultados conhecidos de Teoria de Grupos de Permutações, desvinculada do formalismo clássico Bafna e Pevzner. É importante notar que nosso algoritmo simula, de forma natural, a solução baseada em grafo de ciclos de Bafna e Pevzner. Nossa solução poderá ser automatizada em parte, e acreditamos que indica caminhos novos, que possibilitarão tanto diminuir a razão de aproximação quanto obter uma outra prova usando resultados de Grupos de Permutações para mostrar que o problema da distância de transposição é $\mathcal{NP}$-difícil.

O algoritmo proposto foi implementado na linguagem de programação Java. Utilizamos um sistema de álgebra computacional, chamado GAP, para computar operações envolvendo permutações. O algoritmo foi auditado na ferramenta GRAAu, o que permitiu a comparação de todas as distâncias de transposições dadas por nosso algoritmo, para todas as permutações de tamanho 2 até 11, com os valores exatos. Os resultados dessa auditoria foram comparados com outros encontrados na literatura.

Palavras-chave: Rearranjo de Genomas, Problema da Distância de Transposição, Álgebra, Grupos de Permutações, Algoritmos de aproximação;

Abstract

In computational biology, mutational events affecting large portions of a genome are studied in genome rearrangements. Particularly, *transposition* is a mutational event that changes two contiguous blocks of genes inside a single chromosome. This event generates the *problem of transposition distance*, which is to find the minimum number of transpositions transforming a chromosome into another. Recently, this problem was proved to be $\mathcal{NP}$-hard.

In the literature many algorithms were proposed to solve this problem, taking into account different approaches. In the present work, we will use the algebraic formalism for chromosome modeling and transpositions proposed by Meidanis and Dias, and classic results of Permutation Groups to suggest a 2-approximation algorithm for the transposition distance problem.

Although there are better approximation algorithms, the contribution of this work is the proposition of a solution to the transposition distance problem using only known results of Permutation Groups Theory, dissociated from the classic formalism proposed by Bafna and Pevzner. It is worth noting that our algorithm simulates, in a natural way, the solution based on Bafna and Pevzner's cycle graph. Our solution can be automated in part, and we believe that it indicates new ways that enable to decrease the approximation ratio and to achieve another proof, using results of Permutation Groups, to show that the problem of transposition distance is $\mathcal{NP}$-hard.

The proposed algorithm was implemented using Java programming language. We have used a computer algebra system, called GAP, to compute operations involving permutations. The algorithm was also audited in GRAAu tool, which allowed the comparison of all transposition distances given by our algorithm, for all permutations of size 2 to 11, with the exact values. The results of this audit were compared with others found in the literature.

Keywords: Genome Rearrangement, Transposition Distance Problem, Algebra, Permutation groups, Approximation algorithms;

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Segundo a teoria da evolução de Darwin [17], todas as espécies atuais descendem de espécies ancestrais que sofreram modificações ao longo do tempo, e toda a diversidade biológica atualmente existente decorre do processo de seleção natural. Em Biologia Molecular, essa teoria é levada em consideração quando, através de comparações, buscamos relações de homologia entre sequências genômicas (fragmentos de DNA) de organismos de espécies diferentes. Homologias podem ser inferidas por medidas como similiaridade [61], que identificam quão "parecidas" são duas sequências. O pressuposto básico é o de que sequências genômicas "similares" foram conservadas durante o processo de evolução e possivelmente exercem o mesmo papel nos mecanismos celulares, ou seja, têm a mesma função biológica.

Nas últimas três décadas, projetos de sequenciamento genômico [26, 55] em todo o mundo geraram um enorme volume de dados. Apenas como exemplo, o Projeto Genoma Humano [45] resultou no sequenciamento de $3,2$ bilhões de pares de bases. Seria inviável comparar volumes de dados em tal escala sem uma abordagem automatizada e portanto a utilização de métodos computacionais tornou-se imprescindível à pesquisa em Biologia Molecular. Nesse sentido, para auxiliar a identificação de funções biológicas das sequências genômicas, é fundamental que sejam criados métodos computacionais corretos e eficientes para a tarefa de comparação de sequências. Os resultados das comparações *in silico* potencialmente indicam direções de pesquisa promissoras para os biólogos, que podem então projetar seus experimentos *in vitro* e *in vivo* com base neles.

Pesquisas em Biologia Molecular e a constante busca por novas técnicas computacionais, capazes de analisar a enorme quantidade de informações genômicas disponíveis, são justificadas, pois seus resultados têm utilização em várias aplicações importantes para a sociedade, como a descoberta de novas drogas, aumento da produtividade agrícola, combate de pragas e doenças em espécies importantes para a produção alimentícia ou ainda, investigar o processo de evolução das espécies vivas atualmente.

Os métodos computacionais utilizados na pesquisa genômica utilizam teorias estudadas em diferentes subáreas de Ciência da Computação. Podemos citar, dentre elas: Algoritmos [2, 64], Inteligência Artificial [46, 58] e Sistemas Distribuídos e Paralelos [59, 60, 62]. Esses métodos baseiam-se na comparação de dados genômicos tanto em nível de nucleotídeos [1, 44], quanto em nível de fragmentos de cromossomos [3, 5, 6, 24, 35, 65]. As comparações são efetuadas considerando os eventos que possivelmente tenham ocorrido na natureza [21, 54, 57] e que constituem fatores importantes no processo evolucionário.

Nesta dissertação, o foco é a comparação no nível de fragmentos de cromossomos, pois ela representa o escopo no qual está inserido o problema aqui investigado.

1.1 Rearranjo de Genomas

Esta seção destina-se à apresentação de conceitos básicos de Rearranjo de Genomas e uma breve revisão bibliográfica sobre a área [28, 61].

A comparação entre dois cromossomos é realizada considerando os blocos de genes que ocorrem simultaneamente em ambos e em eventos de mutação conhecidos pelos biólogos, que mostram um cenário plausível de evolução. A construção de tal cenário considera a Hipótese de Parcimônia, a qual sugere que as espécies surgem, ao longo da evolução, a partir de um número mínimo de mutações. Portanto, na área de **Rearranjo de Genomas**, os pesquisadores têm como problema básico o de determinar o menor número de eventos mutacionais, doravante chamados de **eventos de rearranjo** ou simplesmente **rearranjos**, atuando sobre fragmentos de cromossomos com a finalidade de transformar um genoma em outro.

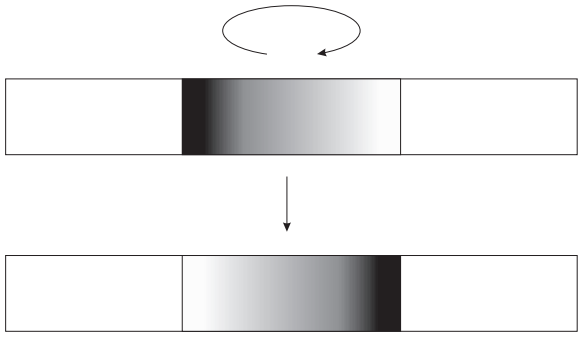

Figura 1.1: Evento de reversão atuando em um cromossomo linear.

Diversos eventos de rearranjo têm sido estudados nesta área. Dentre os mais investigados, podemos citar três eventos agindo num mesmo cromossomo: **reversão**, que atua "invertendo" um fragmento do cromossomo (Figura 1.1); **transposição**, que "corta" um fragmento e o "cola" em outro local (Figura 1.2); e **troca-de-blocos** que "corta" e "troca" de lugar dois fragmentos do cromossomo (Figura 1.3). Podemos citar também *Double-Cut-and-Join* (Figura 1.4) e *Single-Cut-Or-Join* (Figura 1.5), que representam operações de rearranjo genéricas, capazes de simular vários tipos de eventos atuando sobre um ou mais cromossomos.

Cada um dos eventos de rearranjo, assim como cada uma das combinações entre eles, gera um problema de otimização diferente e vários métodos foram propostos para resolver cada um deles.

1.1.1 Conceitos básicos

Nesta seção, mostraremos inicialmente como cromossomos lineares são usualmente modelados em Rearranjo de Genomas. Em seguida, apresentaremos a formalização dos eventos de rearranjo mencionados anteriormente.

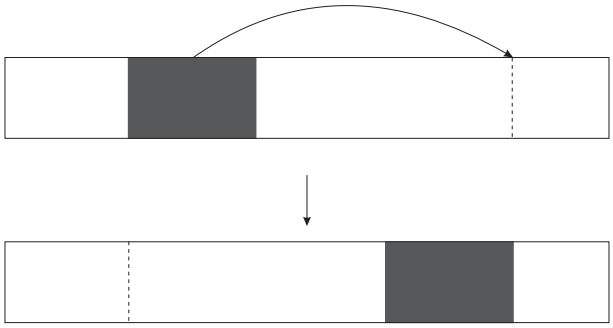

Figura 1.2: Evento de transposição atuando em um cromossomo linear.

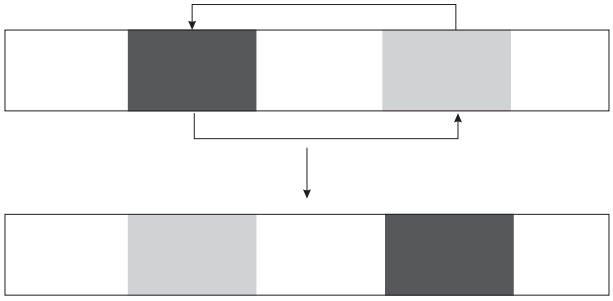

Figura 1.3: Evento de troca-de-blocos atuando em um cromossomo linear.

Modelagem de cromossomos

Em Rearranjo de Genomas, cromossomos normalmente são modelados como permutações de número inteiros. Cada número rotula um fragmento, formado por um ou mais blocos de genes do cromossomo. Eventualmente, esses números podem estar acompanhados de sinais, indicando que a orientação de cada fragmento é conhecida. Se dois fragmentos possuem a mesma orientação, associamos a eles o sinal "+". Caso contrário, associamos ao segundo fragmento o sinal "-".

As permutações sem sinais são representadas por bijeções sobre um conjunto finito $\{1, 2, \ldots n\}$. A representação elementar para uma bijeção é a representação de duas linhas (cada $\pi_i \in \{1, 2, \ldots n\}$ e $1 \leq i \leq n$):

$$\pi = \begin{pmatrix} 1 & 2 & \ldots & n \\ \pi_1 & \pi_2 & \ldots & \pi_n \end{pmatrix}.$$

Porém, em Rearranjo de Genomas, a representação mais comum é a de uma linha, que consiste na representação anterior, porém com a primeira linha suprimida. Desta forma, a permutação anterior é representada apenas como $\pi = (\pi_1 \pi_2 \ldots \pi_n)$.

Por sua vez, permutações com sinais são representadas como bijeções sobre o conjunto $\{-n, \cdots -2, -1, 1, 2, \ldots, n\}$. A seguir, a representação de duas linhas de uma permutação π com sinais (cada $\pi_i \in \{1, 2, \ldots n\}$ e $1 \leq i \leq n$):

$$\pi = \begin{pmatrix} -n & \ldots & -2 & -1 & 1 & 2 & \ldots & +n \\ -\pi_n & \ldots & -\pi_2 & -\pi_1 & \pi_1 & \pi_2 & \ldots & \pi_n \end{pmatrix}.$$

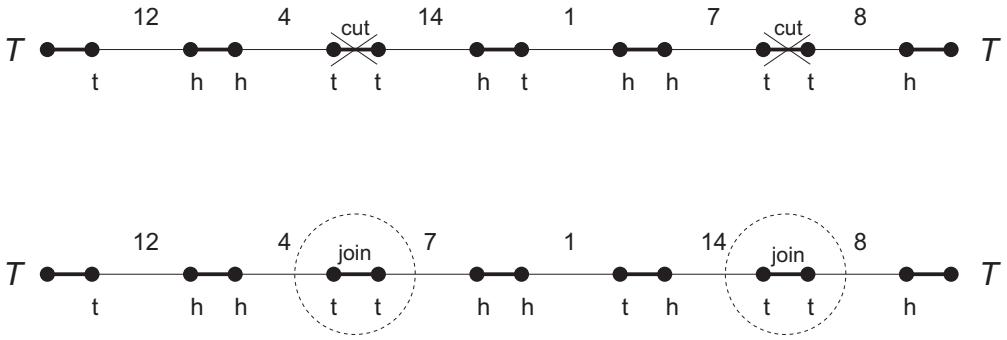

Figura 1.4: Evento DCJ simulando uma reversão (adaptado de Fertin et al. [28]).

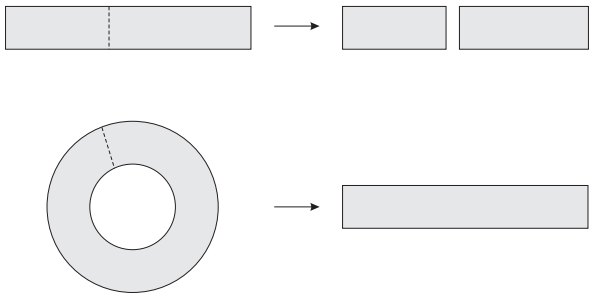

Figura 1.5: Operação *cut* do evento SCJ causando uma fissão em um cromossomo linear e linearizando um cromossomo circular. A operação *join* age de forma contrária.

Na representação mais utilizada, juntamente com a primeira linha, as colunas de domínio negativo também são suprimidas, pois $\pi(-i) = -\pi(i)$. Dessa forma, a representação de permutações com ou sem sinais é semelhante.

Problema da distância de reversão

Definimos uma **reversão** no intervalo $[i, j]$, com $i < j$, denotada por $r[i, j]$, agindo sobre uma permutação $\pi = (\pi_1 \dots \pi_{i-1}\pi_i\pi_{i+1} \dots \pi_{j-1}\pi_j\pi_{j+1} \dots \pi_n)$, por

$$r[i, j] \cdot \pi = (\pi_1 \dots \pi_{i-1}\overline{\pi}_j\overline{\pi}_{j-1} \dots \overline{\pi}_{i+1}\overline{\pi}_i\pi_{j+1} \dots \pi_n),$$

onde $\overline{\pi}_k$ denota a inversão do sinal de $\overline{\pi}_k$.

O **problema da distância de reversão** é formalizado da seguinte forma: dados dois cromossomos, representados pelas permutações π e σ, o problema da distância de reversão entre π e σ é o de encontrar uma sequência de reversões $r_1, r_2, \dots r_\rho$ tal que $r_\rho \dots r_2 \cdot r_1 \cdot \pi = \sigma$, onde ρ é mínimo. Se σ corresponde à permutação identidade $(1\ 2\ \dots\ n)$, chamamos este problema de **problema da ordenação por reversões**. O número ρ é chamado de **distância de reversão** e denotamos a função que toma dois cromossomos, π e σ, e devolve a distância de reversão entre eles, por $d_\rho(\pi, \sigma)$. Se σ é a identidade, essa função é denotada apenas por $d_\rho(\pi)$.

Se as orientações dos blocos de genes de π são desconhecidas, a reversão atua apenas invertendo a ordem dos blocos no intervalo $[i, j]$. Logo, a definição da reversão sobre permutações sem sinais é análoga à definição que acabamos de apresentar.

Exemplo 1.1.1. Sejam as permutações com sinais $\pi = (+1\ -5\ +4\ -3\ +2)$ e $\sigma = (+1\ +2\ +3\ +4\ +5)$. A sequência de reversões $r_1 = r(5,5)$, $r_2 = r(3,3)$ e $r_3 = r(2,5)$ transforma π em σ.

$$\begin{aligned} r_1 \cdot \pi &= r(5,5) \cdot (+1\ -5\ +4\ -3\ \underline{+2}) \\ r_2 \cdot r_1 \cdot \pi &= r(3,3) \cdot (+1\ -5\ \underline{+4}\ -3\ -2) \\ r_3 \cdot r_2 \cdot r_1 \cdot \pi &= r(2,5) \cdot (+1\ \underline{-5\ -4\ -3\ -2}) \\ &= (+1\ +2\ +3\ +4\ +5) = \sigma \end{aligned}$$

Exemplo 1.1.2. Se considerarmos as mesmas permutações do exemplo anterior, porém agora sem sinais, teremos que $\pi = (1\ 5\ 4\ 3\ 2)$ e $\sigma = (1\ 2\ 3\ 4\ 5)$. A reversão $r(2,5)$ é suficiente para transformar π em σ.

Problema da distância de transposição

O evento de **transposição** faz com que blocos de genes de um cromossomo sejam "cortados" e em seguida "colados" numa outra parte do mesmo cromossomo.

De maneira formal, uma transposição atuando sobre um cromossomo representado pela permutação π com n blocos de genes é definida como $\rho(i,j,k)$, com $1 \leq i < j \leq n+1$, $1 \leq k \leq n+1$ e $k \notin [i,j]$. A transposição "corta" os elementos entre π_i e π_{j-1} e os "cola" entre π_{k-1} e π_k, ou seja

$$\begin{aligned} \rho(i,j,k) \cdot \pi = \rho(i,j,k) \cdot (\pi_1 \dots \pi_{i-1}\ \underline{\pi_i \dots \pi_{j-1}}\ \underline{\underline{\pi_j \dots \pi_{k-1}}}\ \pi_k \dots \pi_n) \\ = (\pi_1 \dots \pi_{i-1}\ \underline{\underline{\pi_j \dots \pi_{k-1}}}\ \underline{\pi_i \dots \pi_{j-1}}\ \pi_k \dots \pi_n). \end{aligned}$$

Repare que a transposição tem o mesmo efeito de trocar as posições de dois blocos entre $[i, j-1]$ e $[j, k-1]$.

O problema da distância de transposição é formalizado como segue. Dados dois cromossomos, modelados pelas permutações π e σ, o **problema da distância de transposição** é encontrar uma sequência de transposições $\rho_1, \rho_2, \dots \rho_t$ tal que $\rho_t \dots \rho_2 \cdot \rho_1 \cdot \pi = \sigma$ e t é mínimo. Se σ corresponde à permutação identidade $(1\ 2\ \dots\ n)$, chamamos este problema de **problema da ordenação por transposições**. O número t é chamado de **distância de transposição** e a função que aplicada a duas permutações π e σ, devolve a distância de transposição entre elas, é denotada por $d_t(\pi, \sigma)$. Se σ é a identidade, essa função é denotada apenas por $d_t(\pi)$.

Exemplo 1.1.3. Sejam as permutações $\pi = (1\ 5\ 2\ 4\ 3)$ e $\sigma = (1\ 2\ 3\ 4\ 5)$. As transposições $\rho_1 = \rho(2,4,5)$ e $\rho_2 = \rho(2,4,6)$, aplicadas nesta sequência, transformam π em σ.

$$\begin{aligned} \rho_1 \cdot \pi &= \rho(2,4,5) \cdot (1\ \underline{5\ 2}\ \underline{4}\ 3) \\ \rho_2 \cdot \rho_1 \cdot \pi &= \rho(2,4,6) \cdot (1\ \underline{4\ 5}\ \underline{\underline{2\ 3}}) \\ &= (1\ 2\ 3\ 4\ 5) = \sigma \end{aligned}$$

Problema da distância de troca-de-blocos

O evento de troca-de-blocos pode ser visto como uma generalização da transposição, no sentido de que ela troca dois blocos não necessariamente contíguos de um cromossomo.

Podemos formalizar a troca-de-blocos como segue. Para uma permutação π, a troca de blocos $\beta(i,j,k,l)$ com $1 \leq i < j < k < l \leq n+1$, aplicada a π, troca o intervalo $[i, j-1]$ pelo intervalo $[k, l-1]$, transformando π em $\beta(i,j,k,l) \cdot \pi$.

$$\begin{aligned}\beta(i,j,k,l) \cdot \pi = \beta(i,j,k,l) \cdot (\pi_1 \dots \pi_{i-1} \underline{\pi_i \dots \pi_{j-1}} \pi_j \dots \pi_{k-1} \underline{\pi_k \dots \pi_{l-1}} \pi_l \dots \pi_n) \\ = (\pi_1 \dots \pi_{i-1} \underline{\pi_k \dots \pi_{l-1}} \pi_j \dots \pi_{k-1} \underline{\pi_i \dots \pi_{j-1}} \pi_l \dots \pi_n).\end{aligned}$$

Dados dois cromossomos, modelados pelas permutações π e σ, o **problema da distância de troca-de-blocos** é encontrar uma sequência de troca-de-blocos $\beta_1, \beta_2, \dots \beta_b$ tal que $\beta_b \dots \beta_2 \cdot \beta_1 \cdot \pi = \sigma$ e b é mínimo. Se σ corresponde à permutação identidade $(1\ 2\ \dots\ n)$, chamamos este problema de **problema da ordenação por troca-de-blocos**. O número b é chamado de **distância de troca-de-blocos** e a função que aplicada a duas permutações π e σ, devolve a distância de troca-de-blocos entre elas, é denotada por $bid(\pi, \sigma)$. Se σ é a identidade, essa função é denotada apenas por $bid(\pi)$.

Exemplo 1.1.4. Sejam as permutações $\pi = (1\ 5\ 6\ 8\ 2\ 3\ 7\ 4)$ e $\sigma = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)$. As trocas-de-blocos $\beta(4,5,6,8)$ e $\beta(2,5,6,8)$, aplicadas nesta sequência, transformam π em σ.

$$\begin{aligned}\beta_1 \cdot \pi &= \beta(4,5,6,8) \cdot (1\ 5\ 6\ 8\ 2\ 3\ 7\ 4) \\ \beta_2 \cdot \beta_1 \cdot \pi &= \beta(2,5,6,8) \cdot (1\ 5\ 6\ 7\ 4\ 2\ 3\ 8) \\ &= (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8) = \sigma\end{aligned}$$

Problema da distância de *Double-Cut-and-Join*

Uma operação *Double-Cut-an-Join* (DCJ) atua em duas adjacências ab e cd de um genoma Π e transforma essas adjacências em ac e bd ou ad e bc. Dizemos que DCJ "corta" as adjacências ab e cd e as "junta" em ac e bd (ou ad e bc).

Na Figura 1.6, mostramos um evento DCJ cortando um cromossomo linear em dois lugares. Em seguida, duas extremidades são unidas de modo a produzir um cromossomo circular.

Uma operação DCJ pode, entre outros eventos, simular reversões. Duas operações DCJ consecutivas podem simular transposições e troca-de-blocos.

Tanto genomas, como cromossomos lineares, podem ser comparados no âmbito das operações DCJ, mesmo quando apresentam números diferentes de cromossomos e extremidades. Nesse sentido, para DCJ, não faz sentido a ideia de ordenação, apenas a ideia de distância.

Dados dois genomas Π e Γ, a **distância de *Double-Cut-and-Join*** entre Π e Γ, denotada por $dcj(\Pi, \Gamma)$, corresponde ao número mínimo de operações DCJ necessárias para se transformar Π em Γ.

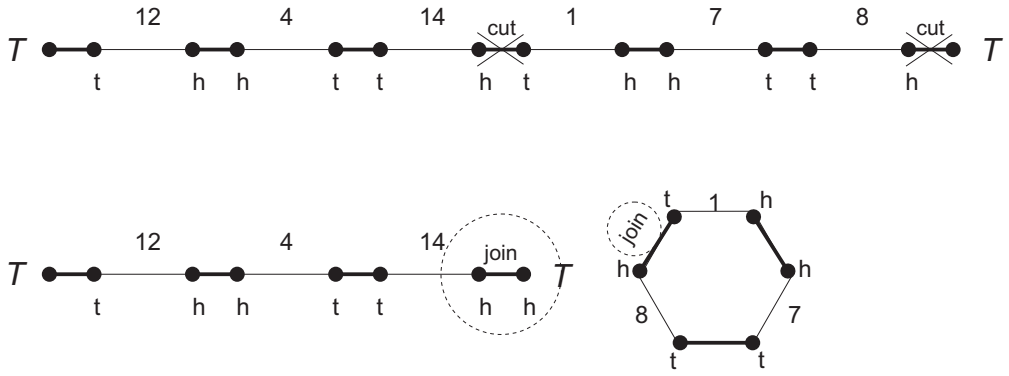

Figura 1.6: Um evento DCJ cortando um cromossomo linear e juntando duas extremidades de modo a produzir um cromossomo circular (adaptado de Fertin et al. [28]).

Problema da distância de *Single-Cut-or-Join*

Em busca de um modelo de rearranjo que fosse o mais elementar possível, Feijão e Meidanis [24] propuseram um modelo chamado *Single-Cut-or-Join* (SCJ). Esse modelo consiste de duas operações básicas: *cut*, que significa "cortar" uma adjacência, e *join*, que significa "juntar" duas extremidades.

Dados dois genomas Π e Γ, a **distância de *Single-Cut-or-Join*** entre Π e Γ, denotada por $d_{SCJ}(\Pi, \Gamma)$, corresponde ao número mínimo de operações SCJ necessárias para se transformar Π em Γ.

1.1.2 Levantamento bibliográfico

Para cada evento de rearranjo mencionado até aqui, vamos discutir brevemente uma revisão dos resultados disponíveis na literatura.

Reversão

O problema da distância de reversão sem sinais é $\mathcal{NP}$-difícil e a sua demonstração é devida a Caprara [13]. Soluções aproximadas para este problema foram apresentadas por Watterson at al. [67], Kececioglu e Sankoff [43], Bafna e Pevzner [5], Christie [15], e Berman, Hannenhalli e Karpinski [10]. Estes últimos forneceram a melhor aproximação até agora conhecida, com razão de 1.375.

Em relação ao problema da distância de reversão com sinais, Hannenhalli e Pevzner [35] foram os primeiros a fornecer uma solução polinomial para este problema, que executa em tempo $O(n^4)$. Este algoritmo foi aperfeiçoado por Tannier, Bergeron e Sagot [63], que forneceram uma solução para o problema em tempo subquadrático $O(n^{3/2}\sqrt{\log n})$. A melhor solução, que executa em tempo linear, é devida a Bader, Moret e Yan [3].

Transposição

Bafna e Pevzner [4] propuseram uma estrutura denominada grafo-de-ciclos para representação de *breakpoints*, que consiste em uma representação dos pontos de ruptura da

Tabela 1.1: Complexidade, método e melhor resultado conhecido para problemas de rearranjo de genomas (adaptado de Feijão e Meidanis [25]).

Eventos de rearranjo	**Complexidade**	**Tipo de algoritmo**	**Autores**
Reversão sem sinais	$\mathcal{NP}$-difícil	Aproximação 1.375	Berman et al. [9]
Reversão com sinais	$O(n)$	Exato	Bader et al. [3]
Transposição	$\mathcal{NP}$-difícil	Aproximação 1.375	Elias e Hartman [23]
Transposição de prefixos	não conhecida	Aproximação 2	Dias e Meidanis [19]
Troca-de-blocos	$O(n^2)$	Exato	Christie [14]
Double-Cut-and-Join	$O(n)$	Exato	Yancopoulos et al. [68]
Single-Cut-or-Join	$O(n)$	Exato	Feijão e Meidanis [24]

ordenação de uma sequência. Levando em conta propriedades dos ciclos dessa estrutura, propuseram algoritmos de aproximação com razões 2, 1.75 e 1.5.

Christie [16], utilizando a estrutura proposta por Bafna e Pevzner, também propôs um algoritmo de aproximação com razão 1.5, porém diferente do proposto por Bafna e Pevzner.

Walter, Dias e Meidanis [66] apresentaram um algoritmo de aproximação quadrático com razão 2.25, baseado em uma estrutura diferente da proposta por Bafna e Pevzner.

Meidanis e Dias [49] formalizaram a teoria de grafo-de-ciclos de Bafna e Pevzner utilizando elementos da Teoria de Grupos de Permutações, o chamado **formalismo algébrico** [49].

Hartman [37] propôs uma variação do grafo-de-ciclos de Bafna e Pevzner, do qual derivou um algoritmo com razão de aproximação 1.5 e complexidade de tempo $O(n^2)$. Posteriormente, utilizando esta estrutura, Elias e Hartman [23] propuseram um algoritmo com razão de aproximação 1.375 e complexidade de tempo $O(n^2)$. Este é o melhor resultado de aproximação conhecido na literatura.

Recentemente, Bulteau, Fertin e Rusu [12] demonstraram que o problema da distância de transposição é $\mathcal{NP}$-difícil.

Uma variante do problema da distância de transposição é o **problema da distância de transposição por prefixos**. Uma transposição por prefixos é uma transposição $\rho(i, j, k)$, onde $i = 1$. Este problema foi introduzido por Dias e Meidanis [19], aos quais também é devido o melhor algoritmo de aproximação, com razão 2 e complexidade de tempo $O(n^2)$. Além disso, Dias e Meidanis mostraram que qualquer transposição pode ser simulada com até duas transposições por prefixos. Portanto, qualquer algoritmo de aproximação com razão k para o problema da distância de transposição é também um algoritmo de aproximação com razão $2k$ para o problema da distância de transposição por prefixos. A complexidade deste problema é desconhecida.

Troca-de-blocos

O problema da ordenação por troca-de-blocos foi introduzido por Christie [14], que também forneceu uma solução em tempo $O(n)$. Se a distância de troca-de-blocos for δ, uma solução alternativa proposta por Lin et al. [47] executa em tempo $O(\delta n)$. Esta solução foi desenvolvida utilizando o formalismo algébrico de Meidanis e Dias [49]. Uma outra solução alternativa foi provida por Feng e Zhu [27], e executa em tempo $O(n \log n)$.

Double-Cut-and-Join

Esta operação foi introduzida por Yancopoulos et al. [68], os quais forneceram uma solução em tempo quadrático.

Posteriormente, Bergeron et al. [7] estenderam a teoria *DCJ*, criando uma estrutura chamada de grafo-de-adjacências, que consiste num grafo gerado pela união de caminhos e ciclos usados na modelagem dos cromossomos. Estes autores desenvolveram um algoritmo guloso utilizando essa estrutura, que executa em $O(n)$.

Mira e Meidanis [53] também apresentaram uma solução linear para este problema, utilizando o formalismo algébrico de Meidanis e Dias [49].

Single-Cut-or-Join

Esta operação foi introduzida recentemente na literatura de Rearranjo de Genomas, e por enquanto contamos apenas o trabalho pioneiro de Feijão e Meidanis [24], cuja solução executa em tempo linear $O(n)$.

1.2 Motivação

Tanto quanto sabemos, predominam nas soluções do problemas em Rearranjo de Genomas, conceitos e argumentos essencialmente gráficos, carecendo de um formalismo mais rigoroso. Como dito antes, Meidanis e Dias [49] propuseram uma teoria de rearranjos nova, baseada na Teoria de Grupos de Permutações, chamado de **formalismo algébrico**. O método proposto consiste na modelagem dos conceitos e argumentos das teorias clássicas de rearranjo em definições e demonstrações algébricas, no sentido de permitir a aplicação direta de resultados já conhecidos da Teoria de Grupos de Permutações.

Vários trabalhos aplicaram esta teoria para desenvolver algoritmos para problemas de rearranjo específicos ou combinações deles. Dentre alguns, podemos citar Feijão e Meidanis [24], Dias e Meidanis [18, 19], Mira et al. [50], e Mira e Meidanis [53], Lin et al. [47], Huang et al. [41]. Embora o formalismo algébrico tenha sido empregado nesses trabalhos, as definições e argumentos das teorias clássicas de rearranjo ainda determinam a forma como os algoritmos são construídos.

1.3 Objetivo

O objetivo deste trabalho é apresentar uma solução para o problema da distância de transposição utilizando apenas resultados da Teoria de Grupos de Permutações.

1.4 Organização dos capítulos

No Capítulo 2, faremos uma revisão sobre conceitos básicos da Teoria de Grupos de Permutações necessários ao entendimento deste trabalho. Apresentaremos também definições e resultados sobre fatoração de permutações.

O Capítulo 3 será destinado a uma revisão dos formalismos clássico [6] e algébrico [49, 51, 52], utilizados na abordagem do problema da distância de transposição.

No Capítulo 4, demostraremos, usando resultados de Grupos de Permutações, limites inferior e superior para o problema da distância de transposição, e proporemos um algoritmo de aproximação com razão 2.

No Capítulo 5, serão discutidos os resultados obtidos pela implementação usando GAP [32], um sistema de álgebra computacional para operações envolvendo permutações. Além disso, apresentamos medidas obtidas pelo GRAAu [30], que é uma ferramenta de auditoria de algoritmos de rearranjo.

Por fim, o Capítulo 6 trará as conclusões e proporá trabalhos futuros.

Capítulo 2

Grupos de Permutações

O objetivo deste capítulo é apresentar definições e resultados de Grupos de Permutações necessários ao entendimento desta dissertação. Na Seção 2.1, veremos conceitos básicos sobre Grupos de Permutações e uma breve introdução ao Grupo Alternado A_n. Em seguida, na Seção 2.1, definiremos produto minimal de 2-ciclos e mostraremos alguns resultados acerca deste conceito. Por fim, na Seção 2.3, apresentaremos uma definição de fatoração de n-ciclos e um lema que será utilizado na demonstração de correção do nosso algoritmo de aproximação.

As demonstrações de lemas e teoremas enunciados neste capítulo foram omitidas. No caso das duas primeiras seções, as demonstrações estão disponíveis em textos básicos de Álgebra. Os mais utilizados neste trabalho foram Herstein [38] e Gallian [29]. As demonstrações de lemas e teoremas enunciados nas seções posteriores estão disponíveis nos artigos referenciados.

2.1 Conceitos básicos

Definição 2.1.1. Dado um conjunto E de elementos, uma **permutação** α de E é uma bijeção arbitrária $\alpha : E \to E$.

Denotamos por $\mathcal{B}(E)$, o conjunto de todas as permutações de E. A composição entre duas permutações de E é novamente uma bijeção de E, assim como o mapeamento identidade $\iota : E \to E, x \mapsto x, \forall x \in E$, e também a inversa α^{-1} de qualquer bijeção $\alpha \in \mathcal{B}(E)$. Além disso, a composição de funções é uma operação associativa. Consequentemente, $\mathcal{B}(E)$ é um grupo, sob a operação de composição de bijeções, denotada por $\circ$.

O grupo $(\mathcal{B}(E), \circ)$ é frequentemente chamado de **grupo simétrico** de E e denotado como $Sym(E)$ (veja por exemplo Garcia e Lequain [33]).

Do ponto de vista algébrico, não importa o tipo dos elementos do conjunto E, somente a cardinalidade de E é relevante. De fato, se E e F são conjuntos da mesma cardinalidade, isto é, se existe uma bijeção $f : E \to F$, então f induz um isomorfismo $\tilde{f} : (\mathcal{B}(E), \circ) \to (\mathcal{B}(F), \circ)$ entre E e F. Em particular, se E é um conjunto finito, digamos com $n \geq 2$ elementos, então assumimos que $E = \{1, 2, \ldots, n\}$. Neste caso, o grupo $Sym(E)$ é também denotado como S_n, o grupo simétrico sobre n símbolos, cuja ordem é $|S_n| = n!$.

Notação 2.1. Pelo fato de S_n ser um grupo multiplicativo, serão recorrentes neste texto, sentenças do tipo: "...multiplicar α por β ..." ou "...produto entre α e β ...". De forma rigorosa, o que se pretende, de fato, nessas sentenças é "compor α com β".

Da mesma forma que para qualquer função definida sobre um conjunto finito, a seguinte notação pode ser usada para representar a permutação α de $E = \{1, 2, \ldots, n\}$:

$$\alpha = \begin{pmatrix} 1 & \ldots & i & \ldots & n \\ \alpha(1) & \ldots & \alpha(i) & \ldots & \alpha(n) \end{pmatrix}$$

Aqui, qualquer outra matriz obtida da matriz acima, representada através do intercâmbio de suas colunas, representará a mesma permutação α. Além disso, como a segunda linha inclui precisamente os elementos de E, a função inversa α^{-1} de α pode ser obtida simplesmente trocando a linha superior pela inferior:

$$\alpha^{-1} = \begin{pmatrix} \alpha(1) & \ldots & \alpha(i) & \ldots & \alpha(n) \\ 1 & \ldots & i & \ldots & n \end{pmatrix}$$

Na sequência, utilizaremos α_x para denotar $\alpha(x)$, para $x \in E$, e $\alpha\beta$ para denotar a composição $\alpha \circ \beta$, que equivale a $(\alpha\beta)(x) = \alpha(\beta(x))$, para todo $x \in E$.

Exemplo 2.1.1.

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 2 & 4 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 4 & 3 & 2 & 5 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 2 & 5 & 1 \end{pmatrix}$$

Para qualquer permutação α de E, um elemento $x \in E$ é chamado de *elemento fixo* de α, se $\alpha(x) = x$. Desta forma, a permutação identidade ι fixa todos os elementos de E.

Uma permutação $\beta \in S_n$ é chamada de **permutação cíclica** ou **ciclo**, se existir um subconjunto $\{a_1, \ldots, a_k\} \subseteq E$ tal que

1. $\beta(a_i) = a_{i+1}$, para $i = 1, \ldots, k-1$
2. $\beta(a_k) = a_1$
3. $\beta(x) = x$, para $x \neq a_i$, $i = 1, \ldots, k$

Na notação usual, tal ciclo é escrito como $\beta = (a_1 \ \ldots \ a_k)$, mas qualquer uma das formas $(a_2 \ a_3 \ \ldots \ a_k \ a_1)$, $(a_3 \ \ldots \ a_k \ a_1 \ a_2)$, $\ldots$, $(a_k \ a_1 \ \ldots \ a_{k-1})$ representa o mesmo ciclo β. Se $k = n$, chamamos $(a_1 a_2 \ \ldots \ a_n)$ de **permutação circular**. O número k corresponde ao **comprimento** do ciclo $\beta = (a_1 \ a_2 \ \ldots \ a_k)$ e neste caso, chamamos β de *k-ciclo*. Símbolos fixados por β, em geral não são representados nesta notação.

Na literatura de Grupos de Permutações, 2-ciclos são comumente chamados de **transposições**. Para evitar confusão entre o significado algébrico e o biológico de transposição, neste trabalho, sempre que mencionarmos "transposição", estaremos nos referindo ao significado biológico. Por sua vez, ciclos de comprimento 2 serão chamados simplesmente de 2-ciclos.

Exemplo 2.1.2. Os 24 elementos do grupo S_4 podem ser separados nas seguintes classes:

1. 1-ciclos: $(1)(2)(3)(4) = (1) = (2) = (3) = (4)$ (a identidade ι);
2. 2-ciclos: $(1\ 2) = (1\ 2)(3)(4)$, etc. (são 6 deste tipo);
3. 3-ciclos: $(1\ 2\ 3) = (1\ 2\ 3)(4)$, etc. (são 8 deste tipo);
4. 4-ciclos: $(1\ 2\ 3\ 4)$, etc. (são quatro 4-ciclos), e;
5. os produtos de dois 2-ciclos: $(1\ 2)(3\ 4)$, $(1\ 3)(2\ 4)$ e $(1\ 4)(2\ 3)$, que não são permutações cíclicas.

Definição 2.1.2. Seja $\beta \in S_n$. Chamamos de **suporte** de β, denotado por $Supp(\beta)$, o subconjunto de E consistido dos símbolos de E que são efetivamente movidos por β.

Definição 2.1.3. Seja $\alpha \in S_n$. A **órbita** de um elemento $x \in E$ em α, denotada por $orb(x, \alpha)$, corresponde ao conjunto $\{y \mid \alpha^k(x) = y,\ k < n\}$. O número de órbitas de α sobre E é denotado por $o(\alpha, E)$. Quando uma órbita contém apenas um elemento, ela é chamada de **órbita trivial**.

Definição 2.1.4. Dizemos que duas permutações $\alpha,\ \beta \in S_n$ são **disjuntas** quando ambas têm órbitas não-triviais disjuntas.

Repare que, na última definição, dizer que α e β são permutações disjuntas, equivale a dizer que os suportes das duas permutações são disjuntos.

Lema 2.1.1. Toda permutação α pode ser escrita como um produto de ciclos disjuntos. Esta representação, chamada de decomposição cíclica de α, é única se desconsiderarmos a ordem dos ciclos.

Exemplo 2.1.3. Seja $\alpha = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 1 & 4 & 6 & 7 & 2 & 5 \end{pmatrix}$. Escrevendo α como um produto de ciclos disjuntos, teremos que $\alpha = (1\ 3\ 4\ 6\ 2)(5\ 7)$. De acordo com o Lema 2.1.1, esta representação é única.

Definição 2.1.5. Sejam α e $\beta \in S_n$. Dizemos que α e β são **conjugados** se existir uma permutação $\gamma \in S_n$ tal que $\gamma\alpha = \beta\gamma$, ou seja $\gamma\alpha\gamma^{-1} = \beta$. Chamamos β de **conjugado de α por γ**, denotado como ${}^{\gamma}\alpha = \beta$.

A operação de conjugação tem a característica de preservar a estrutura da permutação que está sendo conjugada. Nesse sentido, a relação "é conjugado de" corresponde a uma relação de equivalência que particiona o grupo de permutações em classes de permutações com a mesma estrutura de ciclos.

Teorema 2.1.1. Sejam α e $\beta \in S_n$. O conjugado de α por β, dado por ${}^{\beta}\alpha = \beta\alpha\beta^{-1}$, pode ser obtido de α substituindo cada símbolo α_i de α por $\beta(\alpha_i)$.

A operação de conjugação é muito útil quando desejamos reescrever um produto alterando a ordem dos ciclos não-disjuntos (não-comutáveis).

Exemplo 2.1.4. Sejam as permutações $\alpha = (1\ 2\ 3\ 4)(5\ 6)$ e $\beta = (3\ 5)(1\ 2)$. O produto $\alpha\beta$ pode ser reescrito de forma que:

$$\begin{aligned}
\alpha\beta &= \underline{(1\ 2\ 3\ 4)}(5\ 6)(3\ 5)(1\ 2) \\
&= (5\ 6)\underline{(1\ 2\ 3\ 4)}(3\ 5)(1\ 2) \\
&= (5\ 6)^{(1\ 2\ 3\ 4)}(3\ 5)\underline{(1\ 2\ 3\ 4)}(1\ 2) \\
&= (5\ 6)^{(1\ 2\ 3\ 4)}(3\ 5)^{(1\ 2\ 3\ 4)}(1\ 2)\underline{(1\ 2\ 3\ 4)} \\
&= (5\ 6)(4\ 5)(2\ 3)\underline{(1\ 2\ 3\ 4)}
\end{aligned}$$

2.1.1 Grupo Alternado A_n

Dado o k-ciclo $(a_1\ a_2 \dots a_k)$, podemos escrevê-lo (decompô-lo) como o seguinte produto de 2-ciclos:

$$(a_1\ a_k)(a_1\ a_{k-1}) \dots (a_1\ a_2).$$

Se reescrevermos o ciclo acima, iniciando por a_2 e utilizando a mesma fórmula anterior, produziremos um produto de 2-ciclos diferente do anterior. Em qualquer produto obtido dessa forma, podemos também inserir entre quaisquer pares adjacentes de 2-ciclos de um produto obtido através da fórmula anterior, um produto do tipo $(ab)(ab) = \iota$. Ainda assim, o produto de 2-ciclos equivalerá ao k-ciclo $(a_1\ a_2 \dots a_k)$. De fato, todo k-ciclo pode ser escrito como um produto de 2-ciclos, e esse produto não é único.

Uma vez sabendo que toda permutação em S_n corresponde a um produto de ciclos disjuntos, e cada ciclo então pode ser escrito como um produto de 2-ciclos, temos o seguinte resultado.

Teorema 2.1.2. Toda permutação em S_n pode ser escrita como um produto de 2-ciclos.

Exemplo 2.1.5.

$$\begin{aligned}
(1\ 2\ 3\ 4\ 5) &= (1\ 5)(1\ 4)(1\ 3)(1\ 2) \\
(1\ 6\ 3\ 2)(4\ 5\ 7) &= (1\ 2)(1\ 3)(1\ 6)(4\ 7)(4\ 5).
\end{aligned}$$

Embora não sejam únicas, as decomposições das permutações em produtos de 2-ciclos apresentam uma propriedade que é única: a paridade do número de 2-ciclos em cada decomposição.

Teorema 2.1.3. Se uma permutação α pode ser decomposta num produto de um número par (ímpar) de 2-ciclos, então toda decomposição de α num produto de 2-ciclos só pode ter um número par (ímpar) de 2-ciclos.

Definição 2.1.6. Uma permutação que pode ser decomposta num produto de um número par de 2-ciclos é chamada de **permutação par**. De forma análoga, uma permutação decomposta num produto de um número ímpar de 2-ciclos é chamada de **permutação ímpar**.

Uma consequência do Teorema 2.1.3 é que se considerarmos o conjunto de todas as permutações pares de S_n, o produto entre duas permutações desse conjunto também será par. Como resultado, este conjunto forma um subgrupo de S_n.

Teorema 2.1.4. O conjunto das permutações pares em S_n forma um subgrupo de S_n. Este subgrupo é chamado de **grupo alternado** de grau n e é denotado como A_n.

Uma propriedade importante de A_n é que ele pode ser gerado a partir do produto de ciclos de comprimento 3, que por sua vez são pares. O próximo lema, que estabelece este fato, tem fundamental importância neste trabalho, pois como veremos no próximo capítulo, o evento de transposição pode ser modelado através de 3-ciclos com propriedades específicas.

Lema 2.1.2. Seja $n \geq 3$. O grupo A_n é gerado por 3-ciclos.

Demonstração. Todos os 3-ciclos pertencem ao grupo A_n, uma vez que são permutações pares. Seja $\alpha \in A_n$, $\alpha = \beta_{2r} \dots \beta_2 \beta_1$ um produto de um número par de 2-ciclos, que também pode ser escrito como $\alpha = (\beta_{2r}\beta_{2r-1}) \dots (\beta_2\beta_1)$. Além disso, cada produto $(\beta_{i+1}\beta_i)$ é da forma $(a\ b)(a\ b)$, $(a\ c)(a\ b)$ ou $(c\ d)(a\ b)$, onde os símbolos a, b, c e d são distintos entre si. Se calcularmos cada uma das formas possíveis de $(\beta_{i+1}\beta_i)$, teremos: $(a\ b)(a\ b) = \iota$, $(a\ c)(a\ b) = (a\ b\ c)$ e $(c\ d)(a\ b) = (a\ d\ c)(a\ b\ c)$. Fica assim mostrado que a permutação α pode ser escrita como um produto de 3-ciclos. □

2.2 Produtos minimais de 2-ciclos

Nesta seção, introduziremos o conceito de produto minimal de 2-ciclos. Em seguida, apresentaremos um procedimento de redução de produtos de 2-ciclos, que consiste em uma ferramenta útil na obtenção de produtos minimais. Além disso, apresentaremos uma definição de grafo de um produto de 2-ciclos e um resultado importante acerca dessa definição. Este resultado também constitui em outra ferramenta útil na determinação de produto minimais.

Notação 2.2. Para designar produtos de 2-ciclos, utilizaremos letras maiúsculas do alfabeto grego.

Definição 2.2.1. Seja $\alpha \in S_n$. Chamaremos de **produto minimal** de α, um decomposição de α em produtos de $n - r$ 2-ciclos, onde r corresponde ao número de ciclos disjuntos de α.

Exemplo 2.2.1. Seja $\alpha = (5\ 2\ 6)(1\ 3\ 2\ 4)$. São produtos minimais da permutação α: $(5\ 6)(5\ 2)(1\ 4)(1\ 2)(1\ 3)$ e $(1\ 4)(5\ 3)(1\ 5)(5\ 6)(5\ 2)$.

Proposição 2.2.1. Seja $\alpha = (a \dots b \dots c \dots)$. O ciclo α pode ser escrito como um produto minimal da forma $\Sigma(a\ c)(a\ b)$, onde Σ corresponde a um produto arbitrário de 2-ciclos.

Podemos encontrar em Mackiw [48] uma demonstração de que não existe forma mais econômica, no sentido da utilização do menor número possível de 2-ciclos, de decompor uma permutação num produto de 2-ciclos.

Se não levarmos em conta o fato de que algumas decomposições são equivalentes, apenas trocando de lugar fatores comutáveis, o número de produtos minimais de um k-ciclo é k^{k-2}. Este resultado é devido a Dénes [20]. Entretanto, mesmo contando decomposições equivalentes apenas uma vez, o número de tais decomposições pode ser arbitrariamente grande. Mais precisamente, o número de produtos minimais de um k-ciclo é $\frac{1}{2k-1}\binom{3(k-1)}{k-1}$.

Uma demonstração para o teorema que estabelece este resultado pode ser encontrada em Eidswick [22].

O próximo resultado é devido a Biane [11], e vale para decomposições mais gerais. Gewurz e Merola [34] utilizaram este resultado da forma como será enunciado a seguir.

Lema 2.2.1. Se α é uma permutação e $\Pi = (a_1b_1)(a_2b_2)\dots(a_kb_k)$ é um produto minimal de α, então, para cada i, a_i e b_i estão no suporte do mesmo ciclo em α (escrito como um produto de ciclos disjuntos).

Agora, apresentaremos um procedimento de redução de permutações e uma definição de grafo de um produto de 2-ciclos. Essas definições são úteis na obtenção de produtos minimais. Em seguida, apresentaremos uma série de resultados importantes acerca dessas definições. As definições e resultados a seguir são devidos a Higgs [39], embora alguns resultados de seu trabalho sejam equivalentes a resultados anteriores de Dénes [20].

Definição 2.2.2. Seja Π um produto de 2-ciclos e a, b, c, d símbolos distintos de E. Chamamos de **forma a-reduzida** de Π, denotada por Π^a, o produto obtido pela reescrita de Π utilizando as regras seguintes. Se $|\Pi| \leq 1$ ($|\Pi|$ denota a quantidade de 2-ciclos em Π), então $\Pi^a = \Pi$. Se $|\Pi| \geq 2$, então

$$\Pi^a = (\alpha\beta\Sigma)^a = \begin{cases} \alpha(\beta\Sigma)^a & \text{se } a \text{ não ocorre em } \alpha \\ \Sigma^a & \text{se } \alpha = \beta = (a\ b) \\ (b\ c)[(a\ b)\Sigma]^a & \text{se } \alpha = (a\ b) \text{ e } \beta = (a\ c) \\ (b\ c)[(a\ c)\Sigma]^a & \text{se } \alpha = (a\ b) \text{ e } \beta = (b\ c) \\ \beta(\alpha\Sigma)^a & \text{se } \alpha = (a\ b) \text{ e } \beta = (c\ d). \end{cases}$$

com α e β 2-ciclos e Σ um produto de 2-ciclos.

Exemplo 2.2.2. Seja $\Pi = (a\ b)(b\ c)(a\ d)(b\ d)(a\ b)(a\ c)$. Abaixo, as formas reduzidas Π^a e Π^{ab}:

$$\begin{aligned} &\underline{(a\ b)(b\ c)}(a\ d)(b\ d)(a\ b)(a\ c) \\ &(b\ c)\underline{(a\ c)(a\ d)}(b\ d)(a\ b)(a\ c) \\ &(b\ c)(c\ d)\underline{(a\ c)(b\ d)}(a\ b)(a\ c) \\ &(b\ c)(c\ d)(b\ d)\underline{(a\ c)(a\ b)}(a\ c) \\ &(b\ c)(c\ d)(b\ d)(b\ c)\underline{(a\ c)(a\ c)} \\ &\qquad\underline{(b\ c)(c\ d)}(b\ d)(b\ c) = \Pi^a \\ &\qquad(c\ d)\underline{(b\ d)(b\ d)}(b\ c) \\ &\qquad\qquad(c\ d)(b\ c) = \Pi^{ab} \end{aligned}$$

Lema 2.2.2. Se um produto Π é minimal e a não ocorre em Π, então $\Pi(a\ b)$ é minimal.

Lema 2.2.3. Π move a para b se e somente se Π^a é da forma $\Sigma(a\ b)$ (o enunciado foi adaptado de Higgs [39]).

Definição 2.2.3. Se Π é um produto de 2-ciclos, o **grafo** (não-direcionado) de Π é definido da forma como segue. Os vértices são os n elementos de E e as arestas são os fatores de Π. Um vértice incide em uma aresta se e somente se o símbolo correspondente ao vértice ocorre no fator correspondente à aresta.

O fato deste grafo ser não-direcionado decorre da igualdade $(ab) = (ba)$. Múltiplas ocorrências de um mesmo fator em Π dão origem a múltiplas arestas ligando os mesmos vértices correspondentes aos símbolos movidos por este fator. Um exemplo de grafo representando um produto de 2-ciclos pode ser visto na Figura 2.1 (adaptada de Irving e Rattan [42]).

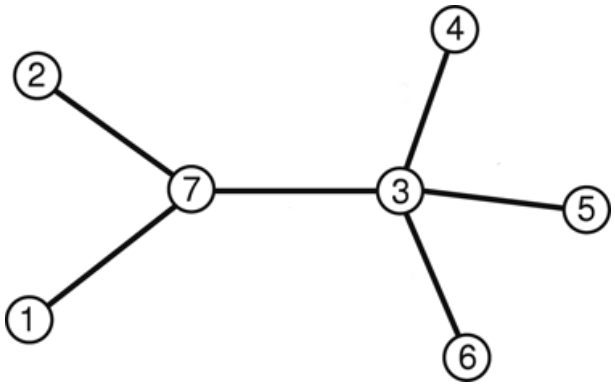

Figura 2.1: Grafo do produto $(3\ 7)(3\ 6)(2\ 7)(3\ 5)(1\ 7)(3\ 4) = (1\ 2\ 3\ 4\ 5\ 6\ 7)$.

Teorema 2.2.1. Seja Π um produto cujo grafo tem k componentes. Π é minimal se e somente se seu grafo é acíclico, i.e, é uma floresta.

Corolário 2.2.1. O grafo de um k-ciclo é uma árvore.

2.3 Fatorações de um n-ciclo

A próxima definição e o lema que segue são devidos a Berkolaiko et al. [8].

Definição 2.3.1. Seja $\beta_m \dots \beta_1$ uma **fatoração** de um n-ciclo $(1\ 2 \dots n)$ em um produto de ciclos menores. Dizemos que uma fatoração é do tipo α se entre os $\{\beta_i\}$ existem exatamente α_2 2-ciclos, α_3 3-ciclos, e assim por diante. Definiremos

$$|\alpha| = \sum_{j\geq 2} \alpha_j,\ \langle\alpha\rangle = \sum_{j\geq 2} (j-1)\alpha_j.$$

A quantidade α satisfaz a relação

$$\langle\alpha\rangle \geq n - 1.$$

Se $\langle\alpha\rangle = n - 1$, dizemos que a fatoração é **minimal**.

Exemplo 2.3.1. Seja o n-ciclo $\beta = (2\ 3\ 5\ 4\ 1)$. São fatorações do tipo 4 de β, portanto minimais: $(2\ 4\ 1)(2\ 3\ 5)$ e $(2\ 3)(4\ 1)(3\ 5\ 1)$.

No próximo lema, $(1\ 2 \ldots n)$ representa uma permutação na qual podemos mapear qualquer outra permutação β sobre um conjunto finito qualquer de símbolos de tamanho n. Escolhemos um símbolo i do suporte de β, então teremos que $i = 1$, $\beta(i) = 2$, $\beta^2(i) = 3$ e assim por diante.

Lema 2.3.1. Seja $\beta = (\beta_1\beta_2 \ldots \beta_k)$, $k \leq n$, um ciclo em uma fatoração minimal do n-ciclo $(1\ 2 \ldots n)$. Então β é crescente $\beta_1 < \beta_2 < \cdots < \beta_k$.

Observe que em cada fatoração minimal no exemplo anterior, os símbolos em cada fator obedecem a propriedade estabelecida no Lema 2.3.1.

Capítulo 3

Formalismos empregados no problema da distância de transposição

Este capítulo tem o objetivo de apresentar breves revisões do formalismo clássico (Seção 3.1) e algébrico (Seção 3.2), empregados na abordagem do problema da distância de transposição.

3.1 Formalismo clássico

Embora o trabalho de Bafna e Pevzner [6] para o problema da distância de transposição tenha resultado em um algoritmo de aproximação de razão 1.5 (esta aproximação foi melhorada posteriormente por Elias e Hartman [23] para 1.375), nesta seção abordaremos apenas as definições fundamentais e os argumentos necessários para a construção do algoritmo de aproximação com razão 2. O intuito é comparar o método proposto neste trabalho com o método empregado no formalismo clássico.

3.1.1 Definições

Vimos no Capítulo 1 que um cromossomo linear com n blocos de genes, onde cada bloco é representado por um número inteiro pertencente ao conjunto $\{1, 2, \ldots, n\}$ é representado como uma permutação $\pi = (\pi_1 \pi_2 \ldots \pi_n)$.

Para definir o conceito de *breakpoint*, adicionaremos dois novos elementos à permutação π: $\pi_0 = 0$ e $\pi_{n+1} = n + 1$, de forma a obter $\pi = (\pi_0 \pi_1 \pi_2 \ldots \pi_n \pi_{n+1})$.

Definição 3.1.1. Para todo i tal que $0 \leq i \leq n$, o par (π_i, π_{i+1}) é um ***breakpoint*** se $\pi_{i+1} \neq \pi_i + 1$ e será denotado na permutação através de um ponto.

Exemplo 3.1.1. Seja a permutação $\pi = (0\ 1\ 3\ 4\ 2\ 5)$. Os *breakpoints* estão representados por um ponto.

$$(0\ 1\ \cdot\ 3\ 4\ \cdot\ 2\ \cdot\ 5)$$

O número de *breakpoints* de uma permutação π é denotado por $b(\pi)$. No exemplo dado, $b(\pi) = 3$. Um algoritmo para o problema de ordenação por transposições deve eliminar todos os *breakpoints* da permutação original, pois a permutação identidade $(0\ 1\ 2\ \ldots\ n)$ é a única que não apresenta *breakpoints*.

Em um evento de transposição, é possível remover ou inserir, no máximo três *breakpoints*. Então, podemos estabelecer o seguinte limite inferior para $d_t(\pi)$:

$$d_t(\pi) \geq \lfloor b(\pi)/3 \rfloor.$$

Este limite não é exato, pois existem permutações que não permitem a diminuição de três *breakpoints* de uma só vez, com uma só transposição, como mostrado no próximo exemplo.

Exemplo 3.1.2. Na permutação $(0 \cdot 4 \cdot 1 \cdot 3 \cdot 2 \cdot 5)$, nenhuma transposição é capaz de diminuir três *breakpoints* de uma só vez. Se aplicarmos a transposição $\rho(1, 2, 5)$ a π, teremos $\rho(1, 2, 5)\pi = (0\ 1 \cdot 3 \cdot 2 \cdot 4 \cdot 5)$. Observe $b(\pi) - b(\rho(1, 2, 5)\pi) = 2$.

Definição 3.1.2. Um **grafo-de-ciclos** de uma permutação π, denotado por $G(\pi)$, é um grafo constituído por um conjunto de vértices $\{0, -1, +1, -2, +2, \ldots, -n, +n, -(n+1)\}$ e um conjunto de arestas direcionadas, onde cada aresta assume uma de duas cores possíveis: preto ou cinza. Cada elemento $\pi_i, 1 \leq i \leq n$ gera dois vértices: $-\pi_i$ e $+\pi_i$. Os elementos $\pi_0 = 0$ e $\pi_{n+1} = n + 1$ geram apenas um vértice cada: $+0$ e $-(n + 1)$, respectivamente. As arestas pretas ligam os vértices $+\pi_i$ e $-\pi_{i+1}$ e são direcionadas no sentido de $-\pi_{i+1}$ para $+\pi_i$. As arestas cinzas ligam os vértices $+\pi_i$ e $-\pi_{i+1}$ e são direcionadas de $+\pi_i$ para $-\pi_{i+1}$.

O grafo-de-ciclos foi proposto com intuito de ser uma representação gráfica de *breakpoints*. Intuitivamente, as arestas pretas indicam a situação atual dos blocos de genes, em relação a sua ordenação no cromossomo, enquanto que as arestas cinzas indicam a situação desejada. A Figura 3.1 mostra $G(\pi)$ para a permutação $\pi = (8\ 5\ 1\ 4\ 3\ 2\ 7\ 6)$.

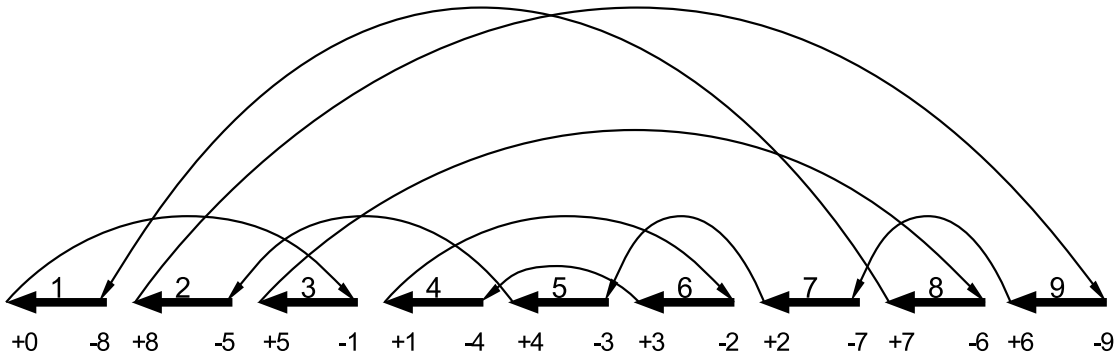

Figura 3.1: Grafo-de-ciclos da permutação $\pi = (8\ 5\ 1\ 4\ 3\ 2\ 7\ 6)$.

Definição 3.1.3. Um **ciclo alternante** de $G(\pi)$ é um ciclo cujas arestas pretas e cinzas se alternam. Para todo vértice de $G(\pi)$, há uma aresta preta saindo e uma cinza incindindo. Isto faz com que a decomposição em ciclos alternantes de $G(\pi)$ seja única. Chamamos de

k-ciclo os ciclos alternantes que têm k arestas pretas. Além disso, chamamos de **longo**, um k-ciclo com $k \geq 3$, e **curto**, caso contrário. Por simplicidade, chamaremos os ciclos alternantes apenas de ciclos.

Tendo em vista que na permutação identidade, para cada aresta preta, existe uma aresta cinza ligando os mesmos vértices, formando um 1-ciclo, a ordenação de uma permutação por transposições deve progressivamente quebrar os ciclos iniciais de $G(\pi)$, até que restem os $n+1$ ciclos da permutação identidade. A Figura 3.2 mostra uma sequência de transposições ordenando $\pi = (4\ 3\ 2\ 1\ 5)$.

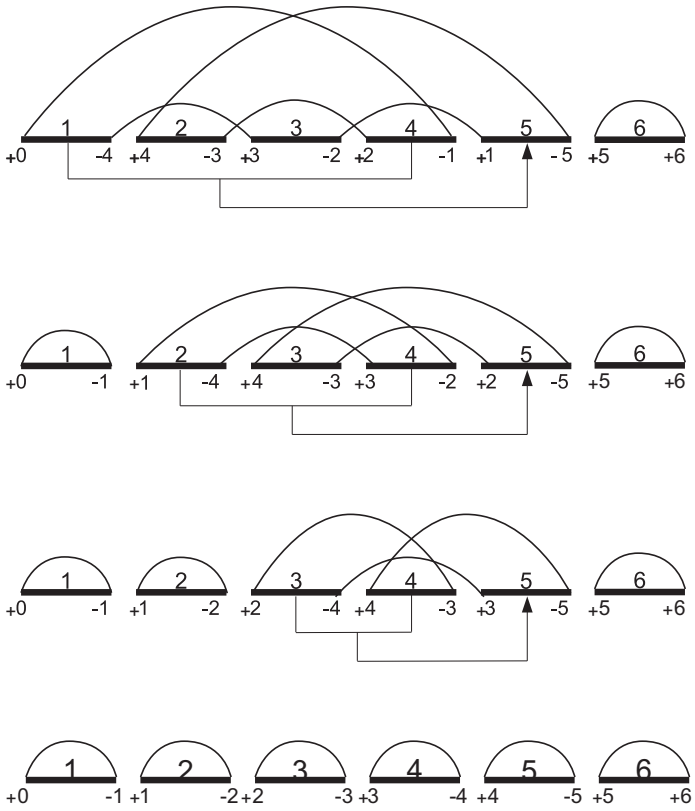

Figura 3.2: Sequência de transposições ordenando $\pi = (4\ 3\ 2\ 1\ 5)$.

3.1.2 Limite inferior

Analisaremos agora como o número de ciclos em um grafo-de-ciclos $G(\pi)$ varia quando aplicamos uma transposição $\rho(i, j, k)$ na permutação π, de forma a gerar um novo grafo-de-ciclos $G(\rho(i, j, k) \cdot \pi)$. Denotaremos o número de ciclos em $G(\pi)$ como $c(\pi)$, e a variação do número de ciclos devido a aplicação da transposição, como $\Delta c(\rho(i, j, k)) = c(\rho(i, j, k) \cdot \pi) - c(\pi)$. Observe que há um abuso de notação, pois embora devolva a variação da quantidade de ciclos de um cromossomo π após a aplicação de uma transposição, a função Δc recebe apenas a transposição aplicada como parâmetro.

Lema 3.1.1. $\Delta(\rho(i,j,k)) \in \{-2,0,2\}$.

Demonstração. A transposição $\rho(i,j,k)$ envolve seis vértices de $G(\pi)$:

$$+\pi_{i-1},\ -\pi_i,\ +\pi_{j-1},\ -\pi_j,\ +\pi_{k-1} \text{ e } -\pi_k.$$

Portanto, a aplicação de $\rho(i,j,k)$ permite a remoção de três arestas pretas, a saber: $(-\pi_i, +\pi_{i+1})$, $(+\pi_j, -\pi_{j-1})$ e $(-\pi_k, +\pi_{k-1})$; e a inclusão de três novas: $(+\pi_j, -\pi_{i-1})$, $(+\pi_i, -\pi_{k-1})$ e $(+\pi_k, -\pi_{j-1})$. As três arestas removidas podem pertencer a um, dois ou três ciclos de $G(\pi)$. Detalharemos e ilustraremos cada uma das situações abaixo. Nas ilustrações, rotularemos os vértices π com suas posições i.

1. Neste caso, as arestas pretas removidas pertencem a três ciclos diferentes de $G(\pi)$. Temos que $c(\rho(i,j,k)\cdot\pi) = c(\pi) - 3 + 1$ (Figura 3.3). Portanto $\Delta c(\pi) = -2$.

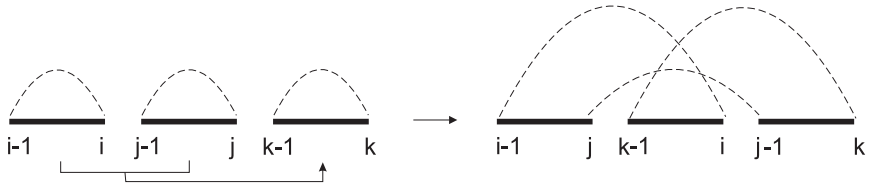

Figura 3.3: $G(\pi)$ contendo três ciclos e $G(\rho(i,j,k)\cdot\pi)$ contendo um ciclo.

2. Neste caso, as arestas pretas removidas pertencem a dois ciclos diferentes de $G(\pi)$. Temos que $c(\rho(i,j,k)\cdot\pi) = c(\pi) - 2 + 2$ (Figura 3.4). Portanto $\Delta c(\pi) = 0$.

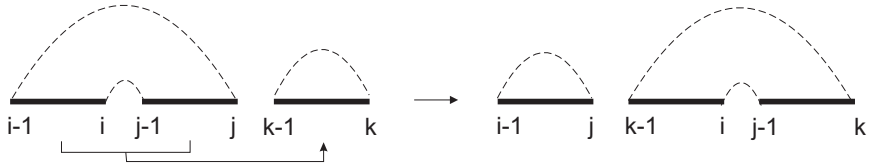

Figura 3.4: $G(\pi)$ e $G(\rho(i,j,k)\cdot\pi)$ contendo dois ciclos.

3. Neste caso, as arestas pretas removidas pertencem a apenas um ciclo diferente de $G(\pi)$. São dois subcasos. Em (a), temos que $c(\rho(i,j,k)\cdot\pi) = c(\pi) - 1 + 1$ (Figura 3.5) e portanto $\Delta c(\pi) = 0$. Em (b), temos que $c(\rho(i,j,k)\cdot\pi) = c(\pi) - 1 + 3$ (Figura 3.6) e portanto $\Delta c(\pi) = 2$

□

Teorema 3.1.1. $d_t(\pi) \geq \frac{(n+1)-c(\pi)}{2}$

Demonstração. Lema 3.1.1. □

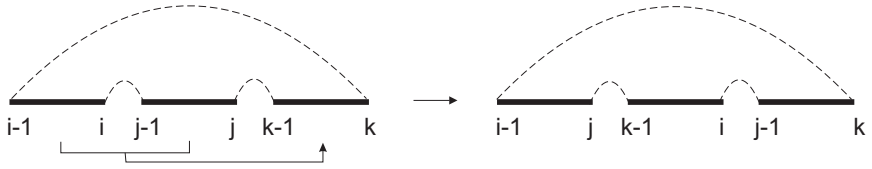

Figura 3.5: $G(\pi)$ e $G(\rho(i,j,k)\cdot\pi)$ contendo um ciclo.

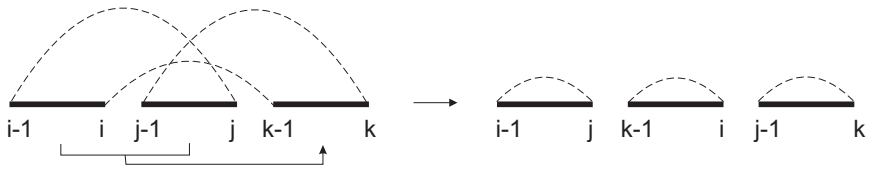

Figura 3.6: $G(\pi)$ contendo um ciclo e $G(\rho(i,j,k)\cdot\pi)$ contendo três ciclos.

Definição 3.1.4. Um ciclo é **ímpar** (**par**) em $G(\pi)$ se ele possui um número ímpar (par) de arestas pretas. Definimos $c_{imp}(\pi)(c_{par}(\pi))$ como sendo o número de ciclos ímpares (pares) em $G(\pi)$.

Podemos notar que todos os ciclos da permutação identidade são ímpares. Assim, precisamos criar o maior número possível de ciclos ímpares. Isso leva a um novo limite inferior.

Para uma permutação π e uma transposição $\rho(i,j,k)$, denotamos $\Delta c_{imp}(\rho(i,j,k)) = c_{imp}(\rho(i,j,k)\cdot\pi) - c_{imp}(\pi)$ como a variação do número de ciclos ímpares ocasionada pela aplicação de $\rho(i,j,k)$ à π.

Lema 3.1.2. $\Delta c_{imp}(\rho(i,j,k)) \in \{-2,0,2\}$.

Demonstração. Pelo Lema 3.1.1, o único caso que leva a criação de mais de dois novos ciclos em $G(\pi)$ é o caso representado na Figura 3.6. Neste caso, três ciclos são adicionados e um ciclo é removido. Se todos os três ciclos adicionados são ímpares, então o ciclo removido é também ímpar, e portanto $c_{imp}(\rho(i,j,k)) = c_{imp}(\pi) - 1 + 3$. Portanto $\Delta c_{imp}(\rho(i,j,k)) \in \{-2,0,2\}$. □

Teorema 3.1.2 (Limite inferior). $d_t(\pi) \geq \frac{(n+1)-c_{imp}(\pi)}{2}$

Demonstração. Lema 3.1.2. □

3.1.3 Limite superior

Definição 3.1.5. Para $x \in \{-2,0,2\}$, um **x-movimento** corresponde a uma transposição $\rho(i,j,k)$ tal que $\Delta c(\rho(i,j,k)) = x$.

Lema 3.1.3. Se a transposição $\rho(i,j,k)$ atua em um ciclo de $G(\pi)$ e cria mais de um novo ciclo, então $\rho(i,j,k)$ é um 2-movimento.

Demonstração. Figura 3.6. □

Lema 3.1.4. Se a transposição $\rho(i, j, k)$ atua em arestas pertencentes a exatamente dois ciclos diferentes de $G(\pi)$, então $\rho(i, j, k)$ é um 0-movimento.

Demonstração. Figura 3.4. □

Definição 3.1.6. Chamamos de **ciclo orientado** o ciclo de $G(\pi)$ que permite a aplicação de 2-movimento e de ciclo **não-orientado**, caso contrário.

Considere um k-ciclo C visitando as arestas pretas $i_1, \ldots, i_k$. O ciclo C pode ser escrito de k formas possíveis, dependendo da escolha da primeira aresta preta. Assumiremos que a sequência começa com a aresta inicial i_1 saindo do vértice mais à direita de π, ou seja, $i_1 = i_t$, com $1 \leq t \leq k$, e t é máximo.

Definição 3.1.7. Uma aresta cinza juntando $\pi_t = i - 1$ com $\pi_s = i$ é chamada de **direcionada à esquerda**, se $t > s$. Caso contrário, é chamada de **direcionada à direita**.

Para todo $k > 1$, um ciclo $C = (i_1, \ldots, i_k)$ é não-orientado se a sequência $i_1, \ldots, i_k$ é decrescente. Caso contrário, C é orientado. Ademais, um k-ciclo C é não-orientado se $k > 1$ e C possuir exatamente uma aresta direcionada à direita.

Lema 3.1.5. Se C é um ciclo orientado, então existe um 2-movimento atuando em C. Se C é um ciclo não-orientado, não existe nenhum 2-movimento atuando sobre C.

Demonstração. Seja $C = (i_1 \ldots i_k)$ um ciclo orientado, e um índice t tal que $3 \leq t \leq k$ e $i_t > i_{t-1}$. A transposição $\rho(i_{t-1}, i_t, i_1)$ gera um 1-ciclo, cujos vértices são $\{\pi_{i_{t-1}-1}, \pi_{i_t}\}$, e outros ciclos. Pelo Lema 3.1.3, $\rho(i_{t-1}, i_t, i_1)$ é um 2-movimento. □

Teorema 3.1.3. Para uma permutação arbitrária π, existe um 2-movimento ou um 0-movimento seguido de um 2-movimento.

Demonstração. Se existe um ciclo orientado em $G(\pi)$, então, pelo Lema 3.1.5, um 2-movimento é possível.

Senão, sejam $C = (i_1 \ldots i_k)$ um ciclo não-orientado em $G(\pi)$, r a posição do elemento máximo de π no intervalo $[i_2, \ldots i_1 - 1]$ e s uma posição $\pi_r + 1$ em π. Claramente $s \notin [\pi_{i_2}, \pi_{i_1}]$. Sem perda de generalidade, assuma que $s > i_1$. Então, a transposição $\rho(r + 1, s, i_2)$ atua nas arestas de dois ciclos diferentes, e portanto, pelo Lema 3.1.4, $\rho(r + 1, s, i_2)$ é um 0-movimento.

Como $\rho(r+1, s, i_2)$ muda a direção da aresta esquerda (π_{i_1-1}, π_{i_2}) e não muda a direção da aresta direita $(\pi_{i_{k-1}}, \pi_{i_1})$, o ciclo C que contém tais arestas em $G(\tau\pi)$ tem pelo menos duas arestas direcionadas à direita e portanto, C é um ciclo orientado e permite um 2-movimento (Figura 3.7).

□

Teorema 3.1.4 (Limite superior). Qualquer permutação π pode ser ordenada em $n + 1 - c(\pi)$ transposições.

Demonstração. Pelo Teorema 3.1.3, $c(\pi)$ cresce, no mínimo, duas unidades a cada duas transposições aplicadas. Na média, um ciclo é criado em $G(\pi)$ a cada transposição. Este teorema representa um limite superior para o problema da distância de transposição. □

Com estes resultados, o seguinte algoritmo pode ser implementado.

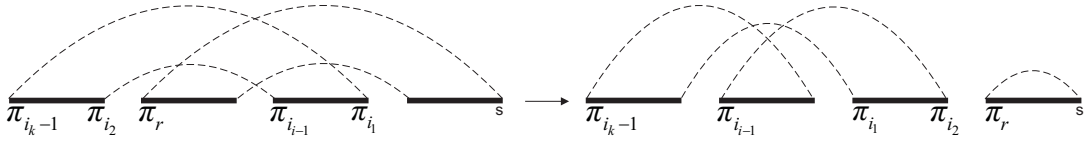

Figura 3.7: Uma transposição agindo em dois ciclos, criando um ciclo orientado.

Algoritmo 1 Algoritmo para o problema da distância de transposição com razão de aproximação 2, segundo método de Bafna e Pevzner.

Entrada: uma permutação π
Saída: lista L contendo uma sequência de transposições capaz de ordenar π

1: crie uma lista vazia L
2: **enquanto** $\pi \neq \iota$ **faça**
3: **enquanto** há ciclos orientados em $G(\pi)$ **faça**
4: $\rho \leftarrow$ 2-movimento ▷ Lema 3.1.3
5: $\pi \leftarrow \rho\pi$
6: adicione ρ no final da lista L
7: **fim enquanto**
8: **se** há k-ciclos em $G(\pi)$, com $k > 1$ **então**
9: $\rho' \leftarrow$ 0-movimento ▷ Teorema 3.1.3
10: $\rho'' \leftarrow$ 2-movimento ▷ Teorema 3.1.3
11: $\pi \leftarrow \rho''\rho'\pi$
12: adicione ρ' no final da lista L
13: adicione ρ'' no final da lista L
14: **fim se**
15: **fim enquanto**
16: **retorne** L

3.2 Formalismo algébrico

Na abordagem clássica, como visto na seção anterior, predominam conceitos e argumentos essencialmente gráficos. No formalismo algébrico proposto por Meidanis e Dias [49], esses conceitos são substituídos por definições algébricas. O intuito foi o de prover uma abordagem mais formal ao estudo dos rearranjos de genomas, e ao mesmo tempo, tirar proveito dos já conhecidos resultados da Teoria de Grupos de Permutações. Neste trabalho, utilizaremos os mesmos conceitos básicos para a modelagem de cromossomos e transposições originalmente introduzidos em Meidanis e Dias [49].

3.2.1 Definições

Definição 3.2.1. A **k-norma** de um permutação π, denotada por $\|\pi\|_k$ corresponde ao número mínimo de k-ciclos no qual π pode ser decomposto. A 2-norma de π é denotada apenas como $\|\pi\|$.

Exemplo 3.2.1. Sejam $\pi = (1\ 6\ 4)(3\ 5\ 2)(0\ 7)$ e $\sigma = (1\ 5\ 3)(0\ 2\ 6\ 4\ 7)$.

$$\pi = (1\ 4)(1\ 6)(3\ 2)(3\ 5)(0\ 7), \|\pi\| = 5$$
$$\sigma = (1\ 3)(1\ 5)(0\ 7)(0\ 4)(0\ 6)(0\ 2), \|\sigma\| = 7$$

Lema 3.2.1. Sejam α e β permutações, temos:

1. $\|\alpha\| = 0$ se e somente se $\alpha = \iota$.
2. $\|\alpha^{-1}\| = \|\alpha\|$
3. $\|\alpha^{\beta}\| = \|\alpha\|$
4. $\|\alpha\beta\| \leq \|\alpha\| + \|\beta\|$
5. $\|\alpha\beta\| = \|\beta\alpha\|$

Definição 3.2.2. Sejam α e β permutações sobre E. Dizemos que α **divide** β se $\|\beta\alpha^{-1}\| = \|\beta\| - \|\alpha\|$. Denotamos esta relação por $\alpha|\beta$.

Exemplo 3.2.2. Sejam $\alpha = (0\ 2\ 1)$ e $\beta = (0\ 4\ 2\ 1\ 3)$. Então, $\alpha|\beta$, pois $\|\beta\alpha^{-1}\| = 2$, $\|\beta\| = 4$ e $\|\alpha\| = 2$.

Definição 3.2.3. O tamanho de um ciclo α corresponde ao tamanho de sua órbita não trivial, ou 0 se $\alpha = \iota$.

Definição 3.2.4. Um ciclo α é dito **ímpar** (**par**) se o seu tamanho é ímpar (par). Denotamos por $o_{impar(par)}(\pi, E)$, o número de órbitas de π de tamanho ímpar (par).

Notação 3.1. Dados um k-ciclo α e uma permutação β, dizemos que um ciclo α "é um ciclo da permutação β", se $\alpha|\beta$ e a órbita não trivial de α pertence ao conjunto de órbitas de β.

Definição 3.2.5. Seja $(x_1\ x_2 \ldots x_n)$ uma permutação sobre o conjunto $E = \{0, 1, 2, \ldots, n\}$ representando um cromossomo linear. No formalismo algébrico, este **cromossomo** será representado pelo $(n+1)$-ciclo $(0\ x_1\ x_2\ \ldots\ x_n)$.

Nota. O símbolo 0 foi incluído para permitir que todas as $n!$ permutações de E possam ser representadas de forma distinta na notação cíclica de permutações.

Definição 3.2.6. Seja π um $(n+1)$-ciclo modelando um cromossomo. Uma transposição é um 3-ciclo $\rho = (u\ v\ w)$, tal que $u, v, w \in E$ e $\rho|\pi$. Neste caso, dizemos que ρ é **aplicável** a π.

Repare que uma transposição no formalismo algébrico está relacionada com os elementos da permutação π e não às posições destes, como na definição de Bafna e Pevzner [6]. A aplicação de uma transposição ρ sobre o cromossomo π, consiste no produto $\rho\pi$.

3.2.2 Limite inferior

Os limites a seguir foram calculados por Mira e Meidanis [51, 52], e são equivalentes aos obtidos por Bafna e Pevzner [6].

O limite inferior foi calculado em função da 3-norma da permutação $\sigma\pi^{-1}$. Repare que se multiplicarmos os dois lados da equação $\rho_t \dots \rho_2\rho_1\pi = \sigma$ por π^{-1}, teremos que $\sigma\pi^{-1}$ equivale à sequência de transposições que transforma π em σ. Além disso, π e σ são permutações que têm o mesmo sinal e portanto, o produto entre elas resulta em uma permutação par, pertencente ao Grupo Alternado A_n, e por consequência, pode ser decomposta como um produto de 3-ciclos.

Repare que se $\sigma = (0\ 1\ \dots n)$, um mapeamento entre $\sigma\pi^{-1}$ e $G(\pi)$ é possível, considerando que as órbitas de $\sigma\pi^{-1}$ correspondem aos ciclos alternantes de $G(\pi)$. Se percorrermos um ciclo alternante de $G(\pi)$, anotando os vértices de onde saem as arestas pretas, a sequência formada corresponde exatamente a uma órbita de $\sigma\pi^{-1}$.

Teorema 3.2.1. Dados dois cromossomos π e σ sobre E, temos que

$$d_t(\pi, \sigma) \geq \|\sigma\pi^{-1}\|_3.$$

Este limite não leva em consideração a paridade dos ciclos de $\sigma\pi^{-1}$ e será melhorado em seguida.

Lema 3.2.2. Se α é uma permutação par sobre E tal que α é um produto de m 3-ciclos $\alpha_m \dots \alpha_2\alpha_1$, então $m \geq \frac{n-o_{impar}(\alpha)}{2}$, com $n = |E|$.

Demonstração. Faremos uma indução em k. Se $k = 0$, então $\pi = \iota$ e $o_{impar}(\pi) = n$. Por consequência, $0 \geq \frac{n-n}{2} = 0$.

Por hipótese de indução, k' é tal que $0 \leq k' < k$ e $\pi = \pi_1 \dots \pi_{k'}$, então $k' \geq \frac{n-o_{impar}(\pi)}{2}$. Pela mesma hipótese, $k \geq \frac{n-o_{impar}(\pi')}{2}$ é verdadeiro para $\pi' = \pi_1 \dots \pi_{k-1}$. Por outro lado, temos que $o_{impar}(\pi) = o_{impar}(\pi'\pi_k) \geq o_{impar}(\pi') - 2$. Então,

$$\begin{aligned} k = (k-1) + 1 &\geq \frac{n - o_{impar}(\pi')}{2} + 1 \\ &= \frac{n - (o_{impar}(\pi') - 2)}{2} \\ &\geq \frac{n - o_{impar}(\pi)}{2} \end{aligned}$$

.

Como consequência, temos que $k \geq \frac{n-o_{impar}(\pi)}{2}$ para $n = |E|$. □

Lema 3.2.3. Se α é uma permutação par sobre E, com $n = |E|$. Então,

$$\|\alpha\|_3 = \tfrac{n-o_{impar}(\alpha)}{2}.$$

Demonstração. Seja um reordenamento dos ciclos da decomposição em ciclos disjuntos de π tal que os ciclos ímpares estão agrupados no início da permutação, i.e., $\pi = \alpha'\alpha''$, onde α' corresponde ao produto de ciclos ímpares de π e α'' ao produto dos ciclos pares de π. Este reordenamento é possível devido a comutatividade dos ciclos disjuntos de uma permutação.

Vamos mostrar que $\|\pi\|_3 \leq \frac{n-o_{impar}(\pi)}{2}$. Para isto, construiremos uma decomposição em 3-ciclos de π através da decomposição em 2-ciclos de α' e α''

Seja $\alpha' = \alpha_1\alpha_2 \ldots \alpha_l$ tal que cada α_i, com $i \in \{1 \ldots l\}$, é um ciclo ímpar de π. Denotamos $n' = |Supp(\alpha')|$ e $n'' = |Supp(\alpha'')|$. Note que $\|\alpha'\| = n' - o_{impar}(\pi)$ e $\|\alpha'\|$ é par, pois a norma de um ciclo ímpar é par. Considere uma decomposição mínima em 2-ciclos de α'. Podemos agrupar os 2-ciclos de α' em pares $\alpha_i\alpha_j$ tais que $|Supp(\alpha_i) \cap Supp(\alpha_j)| = 1$, i.e, os 2-ciclos são da forma $(a\ c)(a\ b)$ e cada um desses pares produz um 3-ciclo $(a\ b\ c)$. Então, $\|\alpha'\|_3 \leq \frac{n'-o_{impar}(\pi)}{2}$.

Agora, considere uma decomposição mínima em 2-ciclos de α''. Tendo que a norma de um ciclo par é ímpar e que $\|\pi\|$ é par, então o número de ciclos $o_{par}(\pi)$ de α'' é par. Seja $L(\alpha_i)$ o tamanho do ciclo α_i, para $i =\in \{l+1 \ldots m\}$. Então, a decomposição em 2-ciclos de α é tal que:

$$\|\alpha''\| = \sum_{i=l+1}^{m} (L(\alpha_i) - 1) = (\sum_{i=l+1}^{m} L(\alpha_i)) - (m - l) = n'' - o_{par}(\pi)$$

Note que existem dois tipos de pares de 2-ciclos: o tipo $\alpha_i\alpha_j$, onde $|Supp(\alpha_i) \cap Supp(\alpha_j)| = 0$; e o tipo $\alpha_r\alpha_s$, onde $|Supp(\alpha_r) \cap Supp(\alpha_s)| = 1$. Chamaremos o primeiro tipo de par de 2-ciclos disjuntos, e o segundo de pares não-disjuntos. Podemos construir um 3-ciclo a partir de um par não-disjunto de forma análoga à que fizemos com pares de 2-ciclos $(a\ c)(a\ b)$ em α'. Por outro lado, um par disjunto $(c\ d)(a\ b)$ gera dois 3-ciclos $(a\ d\ c)(a\ b\ c)$. Então, cada par de 2-ciclos contribui com um 3-ciclo na decomposição em 3-ciclos de α'', mais um número x de 3-ciclos, onde x corresponde ao número de pares disjuntos na decomposição em 2-ciclos de π. Desta forma,

$$\|\alpha''\|_3 = \frac{n'' - o_{par}(\pi)}{2} + x$$

Tendo em vista que os dois ciclos disjuntos estão agrupados aos pares, temos que

$$\lceil \frac{o_{par}(\alpha'') - 1}{2} \rceil \leq x \leq o_{par}(\alpha'') - 1$$

De fato, temos $x = \lceil \frac{o_{par}(\alpha'')-1}{2} \rceil$, pois cada par disjunto gera dois 3-ciclos, pares disjuntos não serão combinados pela paridade do número de 2-ciclos de α''. Mas,

$$\|\alpha''\|_3 = \frac{n'' - o_{par}(\pi)}{2} + \lceil \frac{o_{par}(\alpha'') - 1}{2} \rceil = \frac{n'' - o_{par}(\pi)}{2} + (\frac{o_{par}(\pi) - 2}{2} + 1) = \frac{n''}{2}.$$

Uma vez que α' e α'' são permutações pares e sabendo que a norma de um produto de permutações tem a mesma paridade que a soma das normas das duas permutações, temos então que

$$\|\pi\|_3 \leq \|\alpha'\|_3 + \|\alpha''\|_3 \leq \frac{n' - o_{impar}(\pi)}{2} + \frac{n''}{2} \leq \frac{n - o_{impar}(\pi)}{2}$$

. □

Com esse último resultado, um limite inferior melhor que o anterior, e equivalente ao estabelecido por Bafna e Pevzner [6] pode ser enunciado.

Lema 3.2.4 (**Limite inferior**). Dados dois cromossomos π e σ sobre E, com $n = |E|$. Então,

$$d_t(\pi, \sigma) \geq \frac{n - o_{impar}(\sigma\pi^{-1})}{2}.$$

A seguir, mostraremos os resultados que permitem a implementação de um algoritmo de aproximação com razão 2 utilizando este formalismo.

Definição 3.2.7. Sejam α uma permutação sobre E e $v, w \in E$. Dizemos que w **segue** v para o par (u, α), com $u \in E$, denotado por $v \rightarrow_{u,\alpha} w$, se $v = \alpha^{k_1}u$, $w = \alpha^{k_2}u$, e $0 < k_1 < k_2 < |orb(u, \alpha)|$

Proposição 3.2.1. Sejam um 3-ciclo $\alpha = (u\ v\ w)$ e uma permutação β sobre E. Temos que:

1. $o(\alpha\beta) = o(\beta)$ e $w \rightarrow_{u,\alpha\beta} v$ se e somente se $v \rightarrow_{u,\beta} w$.
2. $o(\alpha\beta) = o(\beta)$, $u, v \notin orb(w, \alpha\beta)$ e $u \in orb(v, \alpha\beta)$ ou $o(\beta\alpha = o(\beta)$, $u, w \notin orb(v, \beta\alpha)$, e $u \in orb(w, \beta\alpha)$ se e somente se $v, w \notin orb(u, \beta)$ e $v \in orb(w, \beta)$.
3. $o(\alpha\beta) = o(\beta) + 2$ e u, v, w pertencem a órbitas distintas em $\alpha\beta$ se e somente se $w \rightarrow_{u,\beta} v$.

Demonstração.

1. Temos que $\alpha\beta = (u \ldots w \ldots v \ldots)$, pois $u, v \in orb(w, \alpha\beta)$. Então $\alpha^{-1}(\alpha\beta) = \iota\beta = \beta$, i.e., $\alpha^{-1}(\alpha\beta) = (w\ v\ u)(u \ldots w \ldots v \ldots) = (u \ldots v \ldots w \ldots) = \beta$. Então, $v \rightarrow_{u,\beta} w$. Por outro lado, se $v \rightarrow_{u,\beta} w$, então $\beta = (u \ldots v \ldots w \ldots)$. Portanto, $\alpha\beta = (u\ v\ w)(u \ldots v \ldots w \ldots) = (u \ldots w \ldots v \ldots)$. Finalmente, temos que $u, w \in orb(v, \alpha\beta)$, $w \rightarrow_{u,\alpha\beta} v$ e $o(\alpha\beta, E) = o(\beta, E) = 1$.
2. Temos que $\alpha\beta = (u \ldots v \ldots)(w \ldots)$, pois $u, v \notin orb(w, \alpha\beta)$ e $u \in orb(v, \alpha\beta)$, então $\alpha^{-1}(\alpha\beta) = \iota\beta = \beta$, i.e., $\alpha^{-1}(\alpha\beta) = (w\ v\ u)(u \ldots v \ldots)(w \ldots) = (w \ldots v \ldots)(u \ldots) = \beta$. Então, $v, w \notin orb(u, \beta)$ e $v \in orb(w, \beta)$. Por outro lado, $\beta = (v \ldots w \ldots)(u \ldots)$, pois $v, w \notin orb(u, \beta)$ e $v \in orb(w, \beta)$. Temos que $\alpha\beta = (u\ v\ w)(v \ldots w \ldots)(u \ldots) = (u \ldots v \ldots)(w \ldots)$. Com isso, $u \in orb(v, \alpha\beta)$ e $u, v \notin orb(w, \alpha\beta)$. Além disso, $o(\alpha\beta, E) = o(\beta, E) = 2$. A outra demonstração da bimplicação é análoga.
3. Temos que $\alpha\beta = (u \ldots)(v \ldots)(w \ldots)$, pois u, v, w pertencem a órbitas diferentes em $\alpha\beta$. Então,$\alpha^{-1}(\alpha\beta) = \iota\beta = \beta$, i.e., $\alpha^{-1}(\alpha\beta) = (w\ v\ u)(u \ldots)(v \ldots)(w \ldots) = (u \ldots w \ldots v \ldots) = \beta$. Temos que $o(\alpha\beta, E) = o(\beta, E)+2$. Portanto, $u, w \in orb(v, \beta)$ e $w \rightarrow_{u,\beta} v$. Por outro lado, temos que $\beta = (u \ldots w \ldots v \ldots)$, pois $w \rightarrow_{u,\beta} v$. Com isso, $\alpha\beta = (u\ v\ w)(u \ldots w \ldots v \ldots) = (u \ldots)(v \ldots)(w \ldots)$. Finalmente, temos que $o(\alpha\beta, E) = o(\beta, E) + 2$ e u, v, w pertencem órbitas distintas em $\alpha\beta$.

□

Lema 3.2.5. Sejam um 3-ciclo $\alpha = (u\ v\ w)$ e uma permutação β sobre E, temos que $\alpha|\beta$ se e somente se $v \rightarrow_{u,\beta} w$.

Demonstração. Se $\alpha|\beta$ então $\|\beta\alpha^{-1}\| = \|\beta\| - \|\alpha\|$. Pela fórmula $\|\alpha\| = |E| - o(\alpha, E)$ e tomando $\|\alpha\| = 2$ temos que $\|\beta\alpha^{-1}\| = |E| - o(\beta\alpha^{-1}, E)$ então $|E| - o(\beta\alpha^{-1}, E) = |E| - o(\beta, E) - 2$. Então, pela Proposição 3.2.1, se u, v, w pertencem a órbitas distintas em $\beta(\alpha^{-1}, v \to_{u,\beta} w$, temos que $o(\beta\alpha^{-1}, E) = o(\beta, E) + 2$. Por outro lado temos que se $v \to_{u,\beta} w$ então $\beta = (u\ v\ w)$. Então, $\beta\alpha^{-1} = (u \dots v \dots w)(w\ v\ u) = (u \dots)(v \dots)(w \dots)$. Sabendo que $o(\beta(\alpha^{-1},\ E) = o(\beta,\ E) + 2$ e $\|\alpha\| = |E| - o(\alpha, E)$ e $\|\alpha\| = 2$, temos que $|E| - \|\beta\alpha^{-1}\| = |E| - \|\beta\| + \|\alpha\| \Rightarrow \|\beta\alpha^{-1}\| = \|\beta\| - \|\alpha\|$, e portanto $\alpha|\beta$. □

Definição 3.2.8. Dadas duas permutações π e σ sobre E. Uma transposição aplicável à π é um x-**movimento** se $o(\sigma\pi^{-1}\tau^{-1}) = o(\sigma\pi^{-1}) + x$.

Lema 3.2.6. Sejam π e σ permutações sobre E. Uma transposição $\rho = (u\ v\ w)$ aplicável à π é um 2-movimento se e somente se $\rho|\sigma\pi^{-1}$.

Lema 3.2.7. Sejam π e σ permutações sobre E. Uma transposição $\rho = (u\ v\ w)$ aplicável à π é um 0-movimento se e somente se $\rho \nmid \sigma\pi^{-1}$ e

1. $\tau^{-1}|\sigma\pi^{-1}$ *ou*
2. $v, w \notin orb(u, \sigma\pi^{-1})$, e $v \in orb(w, \sigma\pi^{-1})$.

Definição 3.2.9. Dadas duas permutações π e σ sobre E. Uma transposição aplicável à π é um x-**movimento válido** se $o_{impar}(\sigma\pi^{-1}\tau^{-1}) = o_{impar}(\sigma\pi^{-1}) + x$.

Definição 3.2.10. Dizemos que a permutação α 3-**divide** uma permutação β, denotado por $\alpha|_3\beta$, se $\|\beta\alpha^{-1}\|_3 = \|\beta\|_3 - \|\alpha\|_3$.

Lema 3.2.8. Sejam π e σ permutações sobre E. Uma transposição $\rho = (u\ v\ w)$ aplicável à π é um 2-movimento válido se e somente se $\rho|_3\sigma\pi^{-1}$.

Demonstração. Se τ é um 2-movimento válido, então $o_{impar}(\sigma\pi^{-1}\tau^{-1}, E) = o_{impar}(\sigma\pi^{-1}, E) + 2$. Como $\|\alpha\|_3 = (|E| - o_{impar}(\alpha)/2$ para uma permutação α sobre E, temos que $|E| - 2\|\sigma\pi^{-1}\tau^{-1}\|_3 = |E| - 2\|\sigma\pi^{-1}\| + 2$. Logo, temos $\|\sigma\pi^{-1}\tau^{-1}\|_3 = \|\sigma\pi^{-1}\|_3 - 1$. Como $\|\tau\|_3 = 1$ para qualquer transposição $\|\sigma\pi^{-1}\tau^{-1}\|_3 = \|\sigma\pi^{-1}\|_3 - \|\tau\|_3$. Então, pela definição de 3-divisibilidade, temos que $\tau|_3\sigma\pi^{-1}$.

Por outro lado, se $\tau|_3\sigma\pi^{-1}$ então $\|\sigma\pi^{-1}\tau^{-1}\|_3 = \|\sigma\pi^{-1}\|_3 - \|\tau\|_3$. Uma vez que $\|\alpha\|_3 = (|E| - o_{impar}(\alpha)/2$, temos que $(|E| - o_{impar}(\sigma\pi^{-1}\tau^{-1}, E))/2 = (|E| - o_{impar}(\sigma\pi^{-1}, E) - 2)/2$. Consequentemente temos que $o_{impar}(\sigma\pi^{-1}\tau^{-1}, E) = o_{impar}(\sigma\pi^{-1}, E) + 2$. Portanto uma transposição aplicável a π é um 2-movimento válido se e somente se $\tau|_3\sigma\pi^{-1}$. □

Definição 3.2.11. Sejam π e σ permutações sobre E. Dizemos que dois ciclos são **ligados** para $\sigma\pi^{-1}$, se $x \to_{u,\pi} \pi\sigma^{-1}u \to_{u,\pi} \sigma\pi^{-1}x$ e u pertence à órbita de α e x pertence à órbita de β.

Lema 3.2.9. Se π e σ são permutações sobre E tais que não existe um 2-movimento aplicável a π, mas há dois ciclos ligados $\alpha = (\pi\sigma^{-1}u\ u \dots)$ e $\beta = (x\sigma\pi^{-1}x \dots)$ para $\sigma\pi^{-1}$ e um 0-movimento $\tau = (u\ x\ \sigma\pi^{-1}\ x)$ aplicável a π, então existe um 2-movimento κ aplicável a $\tau\pi$.

Demonstração. A transposição $\tau = (u\ x\ \sigma\pi^{-1}x)$ é um 0-movimento aplicável a π pela definição de ciclos ligados, Proposição 3.2.1, e Lema 3.2.5. Como α e β são ciclos ligados para $\sigma\pi^{-1}$ então $x \rightarrow_{u,\pi} \pi\sigma^{-1}u \rightarrow_{u,\pi} \sigma\pi^{-1}x$, isto é, temos $x = \pi^{k_1}u$, $\pi\sigma^{-1}u = \pi^{k_2}u$, e $\sigma\pi^{-1}x = \pi^{k_3}u$ (por definição) para $0 < k_1 < k_2 < k_3 < n$ e $n = |\pi|$.

Mostraremos que $\kappa = (u\ \ x\ \pi\sigma^{-1}u)$ é uma transposição aplicável a $\tau\pi$, e que κ é um 2-movimento. Como a transposição troca os blocos consecutivos de x para $\pi^{-1}\sigma\pi^{-1}xe$ de $\sigma\pi^{-1}x$ para u, então, para $\theta = \tau\pi$, temos $x = \theta^{n-k_3+k_1}u$, $\pi\sigma^{-1}u = \theta^{n-k_3+k_2}u$, e $\sigma\pi^{-1}x = \theta^{k_1}u$. Como $k_1 < k_2$, e $n - k_3 > 0$, e $n > k_i$ para $1 \leq i \leq 3$, então $\sigma\pi^{-1}x \rightarrow_{u,\tau\pi} x \rightarrow_{u,\tau\pi} \pi\sigma^{-1}u$. Portanto, existem inteiros m_1 e m_2 tais que $x = \theta^{m_1}u$ e $\pi\sigma^{-1}u = \theta^{m_2}u$, ou, de forma equivalente, $x \rightarrow_{u,\tau\pi} \pi\sigma^{-1}u$. Então, pelo Lema 3.2.5, temos que $(u\ x\ \pi\sigma^{-1}u)|\tau\pi$.

Para mostrar a aplicabilidade de κ sobre $\tau\pi$, e que ele é um 2-movimento, verificamos que $x \rightarrow_{u,\sigma\pi^{-1}\tau^{-1}} \pi\sigma^{-1}u$. Neste caso temos que $\kappa|\sigma\pi^{-1}\tau^{-1}$ pelo Lema 3.2.5, e κ é um 2-movimento aplicável a $\tau\pi$ pelo Lema 3.2.6. Temos então que $\sigma\pi^{-1}\tau^{-1}(\pi\sigma^{-1}u) = u$, pois $\tau^{-1}(\pi\sigma^{-1}u) = \pi\sigma^{-1}u$ e $\sigma\pi^{-1}\pi\sigma^{-1}u = u$. Portanto, se $x \in orb(u, \sigma\pi^{-1}\tau^{-1})$ então $x \rightarrow_{u,\sigma\pi^{-1},\tau^{-1}} \pi\sigma^{-1}u$.

Note que temos $\sigma\pi^{-1}\tau^{-1}(u) = (\sigma\pi^{-1})^2x$ e $\sigma\pi^{-1}\tau^{-1}((\sigma\pi^{-1})^ix) = (\sigma\pi^{-1})^{i+1}x$ para $2 \leq i \leq |orb(x,\ \sigma\pi^{-1}x)| - 1$. Como $(\sigma\pi^{-1})^{|orb(x,\sigma\pi^{-1})|-1}x = (\sigma\pi^{-1})^{-1}x$ então $\sigma\pi^{-1}\tau^{-1}((\sigma\pi^{-1})^{-1}x) = x$. Como consequência, temos que $x = (\sigma\pi^{-1}\tau^{-1})^mu$, para $m = |orb(x,\ \sigma\pi^{-1})|$, e então $x \in orb(u, \sigma\pi^{-1}\tau^{-1})$.

Com tudo isso, temos que $(u\ x\ \pi\sigma^{-1}u)|\sigma\pi^{-1}\tau^{-1}$ e a transposição $(u\ x\ \pi\sigma^{-1}u)$ é um 2-movimento aplicável a $\tau\pi$. □

Proposição 3.2.2 (Bafna e Pevzner [6]). Sejam π e σ permutações sobre E. Se não não um há um 2-movimento aplicável a π, então existem dois ciclos ligados em $\sigma\pi^{-1}$.

O teorema seguinte decorre dos lemas 3.2.6, 3.2.8 e 3.2.9.

Teorema 3.2.2. Sejam π e σ permutações sobre E, então existe um 2-movimento aplicável à π ou um 0-movimento seguido de um 2-movimento.

Com todos esses resultados, o seguinte algoritmo pode ser implementado.

Algoritmo 2 Algoritmo para o problema da distância de transposição com razão de aproximação 2, segundo método introduzido pelo formalismo algébrico.

Entrada: duas permutações π e σ
Saída: lista L contendo uma sequência de transposições capaz de transformar π em σ

1: crie uma lista vazia L
2: **enquanto** $\pi \neq \sigma$ **faça**
3: **se** existe ρ tal que $\rho|\pi$ e $\rho|_3\sigma\pi^{-1}$ **então**
4: $\pi \leftarrow \rho\pi$ ▷ 2-movimento válido
5: adicione ρ no final da lista L
6: **senão**
7: **se** existe ρ tal que $\rho|\pi$ e $\rho|\sigma\pi^{-1}$ **então**
8: $\pi \leftarrow \rho\pi$ ▷ 2-movimento
9: adicione ρ no final da lista L
10: **senão**
11: Sejam $(\pi\sigma^{-1}u\ u\dots)$ e $(x\sigma\pi^{-1}x\dots)$ em $\sigma\pi^{-1}$
12: $\pi \leftarrow (u\ x\ \pi\sigma^{-1}u)\pi$ ▷ 0-movimento
13: adicione ρ no final da lista L
14: **fim se**
15: **fim se**
16: **fim enquanto**
17: **retorne** L

Capítulo 4

Algoritmo de aproximação 2 baseado em Grupos de Permutações

Neste capítulo, vamos propor um algoritmo de aproximação com razão 2 usando conceitos do formalismo algébrico e resultados clássicos de Grupos de Permutações. Na Seção 4.1, demonstraremos limites inferior e superior para o problema da distância de transposição, usando fatoração de n-ciclos conforme apresentado no Capítulo 2. Em seguida, na Seção 4.2, apresentaremos uma prova construtiva, inicialmente proposta para provar a existência das transposições usando resultados de produtos minimais de 2-ciclos.

É importante notar que, embora tenhamos encontrado uma demonstração não construtiva (Seção 4.1) tanto para os limites quanto para a existência das transposições, o arcabouço construído na Seção 4.2 poderá ser utilizado posteriormente para diminuir a razão de aproximação, pois possibilita a geração automatizada dos casos a serem analisados para obter esse novo fator de aproximação.

4.1 Limites para o problema da distância de transposição

No formalismo algébrico original, um 3-ciclo seria aplicável a um cromossomo, representado por um $(n+1)$-ciclo, se este fosse divisível pelo 3-ciclo. Neste capítulo, redefiniremos esse conceito, considerando apenas a ordem em que os símbolos aparecem nas duas permutações.

Definição 4.1.1. Sejam ρ e π, um 3-ciclo e um n-ciclo, respectivamente. Suponha que $\rho = (\pi_i\ \pi_j\ \pi_k)$. Dizemos que ρ é **aplicável** a π se existirem inteiros l e m tais que $\pi^l(\pi_i) = \pi_j$ e $\pi^m(\pi_i) = \pi_k$, com $1 \leq l < m \leq n$.

Note que, pela definição, se $\rho = (\pi_i\pi_j\pi_k)$ é aplicável a π, então o n-ciclo π é da forma:

$$(\pi_1 \dots \pi_{i-1}\pi_i\pi_{i+1} \dots \pi_{j-1}\pi_j\pi_{j+1} \dots \pi_{k-1}\pi_k\pi_{k+1} \dots \pi_n).$$

Se calcularmos o produto $\rho\pi$, obteremos um n-ciclo π' tal que os símbolos entre π_i e π_{j-1} (incluindo estes) em π são recortados e colados entre π_{k-1} e π_k, simulando, portanto, uma transposição biológica em π.

Exemplo 4.1.1. A transposição representada pelo 3-ciclo (1 5 4) atua sobre o cromossomo representado pelo $(n+1)$-ciclo (0 1 3 5 2 4 6), como abaixo:

$$(1\ 5\ 4)(0\ 1\ 3\ 5\ 2\ 4\ 6) = (0\ 5\ 2\ 1\ 3\ 4\ 6).$$

O próximo resultado mostra que em um produto de dois n-ciclos α e β, sempre existe uma tripla (a, b, c) que está invertida em β com relação ao produto $\alpha\beta$, quando esse produto é um k-ciclo.

Proposição 4.1.1. Sejam α e β dois n-ciclos, tais que $\alpha\beta$ é um k-ciclo. Então, existe pelo menos uma tripla (a, b, c) tal que $\beta = (a \dots c \dots b \dots)$ e $\alpha\beta = (a \dots b \dots c \dots)$.

Demonstração. Suponha, por absurdo, que a tripla (a, b, c) enunciada não exista. Desta maneira, teremos que $\alpha\beta$ corresponde a um ciclo cujos símbolos estão em uma sequência crescente, considerando a ordem dos símbolos em β . Neste caso, $\alpha\beta$ seria um fator C_1 de uma fatoração minimal (veja Lema 2.3.1) $C_r \dots C_2 C_1$ de β. Portanto, a única permutação que multiplicada por β resultaria em C_1 seria o inverso de $C_r \dots C_2$, e assim α não poderia ser um n-ciclo. □

O próximo resultado mostra que, quando o produto $\alpha\beta$ de dois n-ciclos α e β não é um k-ciclo e, além disso, não existe uma tripla como a descrita na proposição anterior em um dos ciclos da decomposição disjunta de $\alpha\beta$, sempre existe uma quádrupla (a, b, c, d) tal que $\{a, b\}$ e $\{c, d\}$ estão em suportes de ciclos diferentes de $\alpha\beta$ e esses símbolos estão intercalados em β.

Proposição 4.1.2. Sejam α e β dois n-ciclos, tais que $\alpha\beta = C_r \dots C_2 C_1$ não é um k-ciclo. Se não existe uma tripla (a, b, c) tal que algum ciclo $C_i = (a \dots c \dots b \dots)$ e $\beta = (a \dots b \dots c \dots)$, então existe uma quádrupla (a, b, c, d) tal que $\beta = (a \dots c \dots b \dots d \dots)$ e existem ciclos $C_l = (a \dots b \dots)$ e $C_m = (c \dots d \dots)$, com $l \neq m$ e $l, m \in \{1, 2, \dots r\}$.

Demonstração. Suponha, por absurdo, que a quádrula enunciada não exista. Então, para toda quádrupla de símbolos (a, b, c, d), onde $\{a, b\}$ está em um ciclo C_l e $\{c, d\}$ em um ciclo C_m, onde $l \neq m$ e $l, m \in \{1, 2, \dots k\}$, teremos que $\beta = (a \dots b \dots c \dots d \dots)$. Sem perda de generalidade, o caso em que $\beta = (a \dots c \dots d \dots b \dots) = (b \dots a \dots c \dots d \dots)$ é equivalente ao anterior, bastando trocar os símbolos a e b.

Então, cada ciclo da decomposição disjunta de $\alpha\beta$ corresponde a um ciclo cujos símbolos estão em uma sequência crescente (considerando a ordem dos símbolos em β). Vamos chamar $\alpha\beta$ de γ. Então temos que

$$\begin{aligned} \alpha\beta &= \gamma \\ \beta &= \alpha^{-1}\gamma. \end{aligned}$$

Neste caso, temos que α^{-1} é equivalente a um produto $\tau\upsilon$, onde a permutação υ corresponde a um produto de 2-ciclos capazes de "unir" os ciclos de γ. Se os ciclos forem unidos de forma a manter uma sequência crescente, o produto $\upsilon\gamma$ corresponde a um fator de uma fatoração mínima de β (veja Lema 2.3.1). Finalmente, para que $\tau\upsilon\gamma$ se iguale a β, τ deve corresponder aos outros fatores dessa fatoração de β.

Seja r o número de ciclos na decomposição disjunta de γ. Então, υ seria equivalente a um produto de $(r-1)$ 2-ciclos. Como cada 2-ciclo deste produto une dois ciclos disjuntos,

$\|\upsilon\gamma\| = (r-1) + \|\gamma\|$. Uma vez que $\upsilon\gamma$ é um fator de β, então $\|\tau\| = \|\beta\| - \|\upsilon\gamma\|$. Portanto, como $\alpha^{-1} = \tau\upsilon$, temos que

$$\begin{aligned}\|\alpha^{-1}\| &\leq \|\tau\| + \|\upsilon\| \\ &\leq (\|\beta\| - \|\upsilon\gamma\|) + \|\upsilon\| \\ &\leq ((n-1) - ((r-1) + \|\gamma\|)) + (r-1) \\ &\leq (n-1) - (r-1) - \|\gamma\| + (r-1) \\ &\leq (n-1) - \|\gamma\|.\end{aligned}$$

Sendo $\|\alpha^{-1}\| \leq (n-1) - \|\gamma\|$ e $\|\gamma\| > 0$, então $\|\alpha^{-1}\| < n-1$ e portanto, α não pode ser um n-ciclo.

□

Vamos, agora, aplicar as duas proposições anteriores ao problema da distância de transposição. O resultado seguinte trata do caso em que $\sigma\pi^{-1}$ é um k-ciclo.

Lema 4.1.1. Sejam π e σ cromossomos sobre E. Se $\sigma\pi^{-1}$ é um k-ciclo, com $k \leq n$, então existe um produto minimal de $\sigma\pi^{-1} = \Sigma(a\ c)(a\ b)$, tal que o 3-ciclo $(a\ b\ c) = (a\ c)(a\ b)$ é aplicável a π e Σ é um produto arbitrário de 2-ciclos.

Demonstração. Pela Proposição 4.1.1, existe uma tripla (a, b, c) tal que

$$\sigma\pi^{-1} = (a \ldots b \ldots c \ldots) \text{ e } \pi^{-1} = (a \ldots c \ldots b \ldots).$$

Portanto, o 3-ciclo $(a\ b\ c)$ é aplicável à π, pois $\pi = (a \ldots b \ldots c \ldots)$ e $\sigma\pi^{-1}$ pode ser escrito como o produto minimal $\Sigma(a\ c)(a\ b)$. □

O próximo lema representa uma heurística capaz de melhorar a performance (em relação às métricas do GRAAu [30]) do algoritmo proposto nesta dissertação, conforme resultados descritos no Capítulo 5.

Lema 4.1.2. Sejam π e σ cromossomos sobre E. Se $\sigma\pi^{-1}$ é um k-ciclo, com $k \leq n$, da forma $(a\ b\ c \ldots)$ e $\pi = (a \ldots b \ldots c \ldots)$. Então, existem dois 3-ciclos aplicáveis em sequência à π.

Demonstração. Temos que o 3-ciclo $\rho = (a\ b\ c)$ é aplicável a π. Sendo assim, temos que $\sigma\pi^{-1}\rho^{-1}$ é um $(k-2)$-ciclo e portanto, pelo Lema 4.1.1, existe outro 3-ciclo aplicável a $\rho\pi$. □

O resultado seguinte trata do caso em que $\sigma\pi^{-1}$ não é um k-ciclo.

Lema 4.1.3. Sejam π e σ cromossomos sobre E, tais que $\sigma\pi^{-1}$ não é um k-ciclo. Então:

(1) existe um produto minimal de $\sigma\pi^{-1} = \Delta(a\ c)(a\ b)$ e $(a\ b\ c) = (a\ c)(a\ b)$ é aplicável a π; ou

(2) existe um produto minimal de $\sigma\pi^{-1} = \Delta(c\ d)(a\ b)$, tal que $\pi = (a \ldots d \ldots b \ldots c \ldots)$. Neste caso, os 3-ciclos $(a\ b\ c)$ e $(a\ d\ c)$, gerados a partir do produto $(c\ d)(a\ b) = (a\ d\ c)(a\ b\ c)$, são aplicáveis em sequência a π.

Demonstração.

(1) Se algum ciclo da decomposição disjunta de $\sigma\pi^{-1}$ for da forma $(a \dots b \dots c \dots)$ e o 3-ciclo $(a\ b\ c)$ for aplicável a π, então $\sigma\pi^{-1}$ pode ser escrito da forma $\Delta(a\ c)(a\ b)$.

(2) Pela Proposição 4.1.2, a decomposição disjunta $C_r \dots C_2 C_1$ de $\sigma\pi^{-1}$ tem pelo menos dois ciclos C_i e C_j tais que $C_i = (a \dots b \dots)$ e $C_j = (c \dots d \dots)$ e $\pi = (a \dots d \dots b \dots c \dots)$. Então, $\sigma\pi^{-1}$ pode ser escrito de forma minimal $\Delta(c\ d)(a\ b)$.

□

Os lemas 4.1.1 e 4.1.3 demonstram o seguinte teorema.

Teorema 4.1.1. Sejam π e σ cromossomos sobre E. Sempre existe um 3-ciclo aplicável a π.

O limite inferior decorre do Lema 4.1.1, como mostrado abaixo.

Teorema 4.1.2 (Limite inferior). $d_t(\pi, \sigma) \geq \frac{\|\sigma\pi^{-1}\|}{2}$

Demonstração. Se todos os 3-ciclos que transformam π em σ forem dados pelo Lema 4.1.1 ou pela parte (1) do Lema 4.1.3, teremos que o número de 3-ciclos aplicáveis que transformam π em σ será dado por $\|\sigma\pi^{-1}\|/2$. □

Repare que o limite inferior estabelecido pelo Teorema 4.1.1 equivale àquele do Teorema 3.2.1, no Capítulo 3, pois $\frac{\|\sigma\pi^{-1}\|}{2} = \|\sigma\pi^{-1}\|_3$. Este limite pode ser melhorado considerando apenas os ciclos pares de $\sigma\pi^{-1}$, como já demonstrado no Lema 3.2.2.

O teorema seguinte mostra um limite superior que decorre imediatamente do caso (2) do Lema 4.1.3.

Teorema 4.1.3 (Limite superior). $d_t(\pi, \sigma) \leq \|\sigma\pi^{-1}\|$

Demonstração. Se todos os 3-ciclos que transformam π em σ forem dados pela parte (2) do Lema 4.1.3, teremos que o número de 3-ciclos aplicáveis será $\|\sigma\pi^{-1}\|$, pois para cada par de 2-ciclos de um produto minimal de $\sigma\pi^{-1}$, teremos dois 3-ciclos na sequência de transposições que transforma π em σ. □

4.2 Uma demonstração construtiva de limites para o problema da distância de transposição

Nesta seção, apresentamos uma demonstração construtiva dos limites obtidos na seção anterior. Observamos que na definição seguinte, usamos como hipótese que um 3-ciclo ρ_i de uma sequência de transposições $\rho_r \dots \rho_{i+1}\rho_i \dots \rho_1$, transformando um cromossomo inicial π em um cromossomo final σ, não se cancela no produto $\rho_r \dots \rho_{i+1}$. Além disso, símbolos que não estão no suporte de $\sigma\pi^{-1}$ não serão utilizados. Isso é justificado, pois em nossa teoria, não utilizar símbolos que não estão no suporte de $\sigma\pi^{-1}$ é equivalente, no formalismo clássico, a não aplicar transposições em elementos adjacentes em π (elementos que não apresentam *breakpoints* em relação a σ).

Definição 4.2.1. Dizemos que um 3-ciclo $\beta = (a\ b\ c)$ é **supérfluo** no produto $\alpha\beta$, se $\{a, b, c\} \not\subset Supp(\alpha\beta)$ ou se $\|\alpha\beta\| < \|\alpha\|$. Caso contrário, dizemos que β é **não-supérfluo**.

O próximo resultado é um lema técnico, e destina-se a auxiliar a demonstração do lema subsequente.

Lema 4.2.1. Seja um 3-ciclo $\delta = (a\ b\ c) = (a\ c)(a\ b)$. Seja também um produto de 2-ciclos Π tal que $|Supp(\Pi) \cap \{a, b, c\}| = 2$, e cada um dos dois símbolos dessa intersecção está no suporte de um ciclo diferente na decomposição disjunta de Π. Temos então que δ é não-supérfluo em $\Pi\delta$ e existe um produto minimal de $\Pi\delta$ tal que os dois fatores mais à direita são $(a\ c)(a\ b)$.

Demonstração. Seja Π' um produto minimal de Π. A multiplicação de Π' por δ tem o efeito de adicionar duas arestas ao grafo de Π'. Uma primeira aresta conecta duas árvores do grafo, as árvores que contém os vértices a e c, respectivamente. A segunda aresta conecta as árvores que contém a e b. Desta forma, o grafo de $\Pi'\delta$ é uma floresta e portanto, pelo Teorema 2.2.1, corresponde a um produto minimal. Sendo um produto minimal, então $\|\Pi'\delta\| = \|\Pi\| + 2$. Portanto, δ é não-supérfluo em $\Pi\delta$. □

Notação 4.1. Por simplicidade, na demonstração do próximo resultado, se um símbolo a estiver no suporte uma permutação arbitrária α, diremos apenas que "a está em um ciclo de α". Se soubermos precisar qual é o ciclo em cujo suporte o símbolo a está e, supondo que este ciclo é o ciclo β, diremos apenas "a está em β".

No próximo lema, observamos que Δ designa um produto arbitrário de 2-ciclos e não deve ser confundido com a notação de variação de ciclos do Capítulo 3.

Lema 4.2.2. Seja Γ um produto da forma $\Delta(a\ b\ c)$, onde Δ corresponde a um produto arbitrário de 2-ciclos. Se $(a\ b\ c)$ é não-supérfluo em Γ, então existe um produto minimal Ω de Γ que assume uma das seguintes formas, onde Θ é um outro produto arbitrário de 2-ciclos, equivalente ou não a Δ:

1. $\Theta(a\ c)(a\ b)$; ou
2. $\Theta(a\ b)(a\ c)$; ou
3. $\Theta(b\ d)(a\ c)$; ou
4. $\Theta(c\ d)(a\ b)$; ou
5. $\Theta(a\ d)(b\ c)$.

Demonstração. Se computarmos a forma a-reduzida de Δ, obteremos um produto da forma $\Delta'\phi$, onde $\phi = \iota$, no caso em que o símbolo a desaparece, ou $\phi = (a\ d)$, com $d \neq a$. Note que neste caso, d pode corresponder aos símbolos b ou c. De forma semelhante, se calcularmos a forma b-reduzida de Δ', obteremos um produto do tipo $\Delta''\mu$ tal que, de forma análoga, $\mu = \iota$ ou $\mu = (b\ e)$. Note que neste caso, o símbolo e não pode ser b e também não pode ser a, pois a já não ocorria em Δ'. Se finalmente calcularmos a forma c-reduzida de Δ'', obtendo um produto do tipo $\Delta'''\psi$ tal que $\mu = \psi$ ou $\psi = (c\ f)$, com $f \notin \{a, b, c\}$, então poderemos escrever Δ como $\Delta'''\psi\mu\phi$. A Figura 4.1 ilustra o processo de obtenção dos produtos assumindo esta forma.

Agora, em cada valor possível de $\Delta'''\phi\mu\psi$ (correspondente a cada uma das folhas da árvore na Figura 4.1), substituiremos os símbolos d e e pelos possíveis valores que estes podem assumir, lembrando que $d \neq a$, $e \notin \{a, b\}$ e $f \notin \{a, b, c\}$. Em seguida,

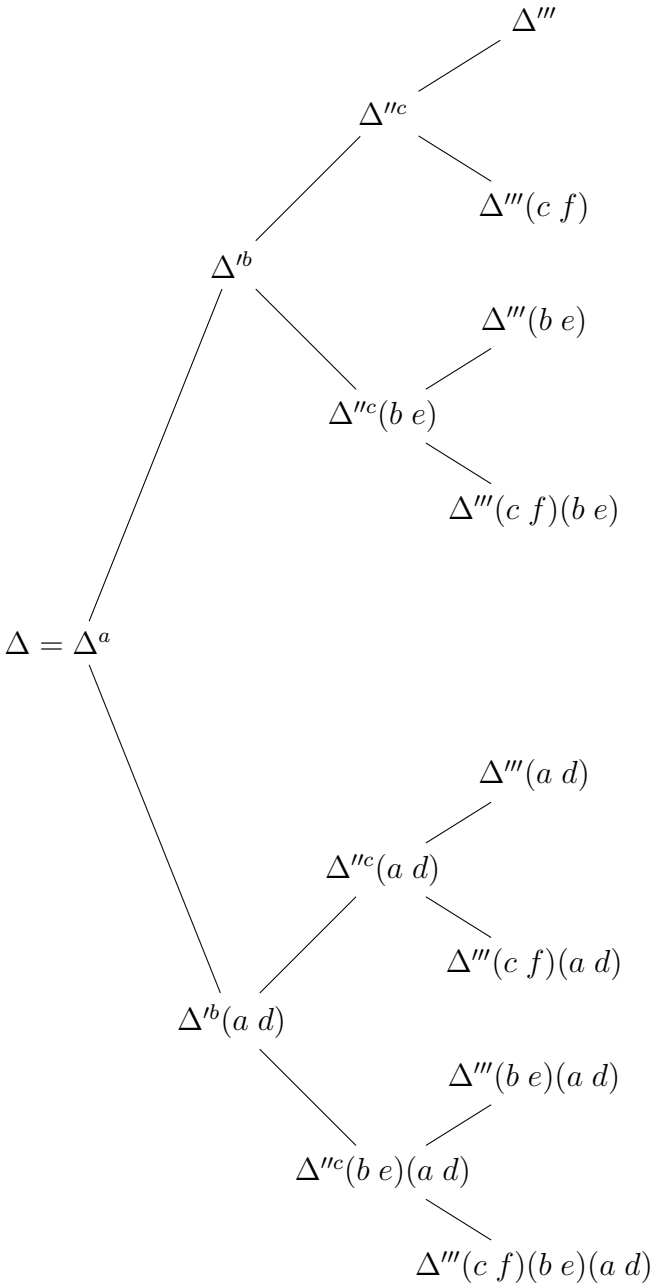

Figura 4.1: Obtenção das possíveis formas $\Delta'''\phi\mu\psi$ de Δ. Os símbolos d, e, f são tais que $d \neq a$, $e \notin \{a, b\}$ e $f \notin \{a, b, c\}$.

multiplicaremos cada produto por $(a\ c)(a\ b) = (a\ b\ c)$, e mostraremos que nos casos em que $\|\Delta'''\phi\mu\psi\| \geq \|\Delta'''\phi\mu\psi(a\ c)(a\ b)\|$ e que $\{a,b,c\} \subset Supp(\Delta'''\phi\mu\psi(a\ c)(a\ b))$, o produto $\Delta'''\phi\mu\psi(a\ c)(a\ b)$ poderá ser reescrito como uma das formas enunciadas neste lema.

Suponha que a decomposição disjunta de $\Delta'''\phi\mu\psi$ seja $\alpha_1\alpha_2\ldots\alpha_m$.

1. Δ'''. O produto $\Delta'''(a\ c)(a\ b)$ pode trivialmente ser escrito como um produto minimal. Suponha um produto minimal ψ de Δ'''. Pelo Lema 2.2.2, $\psi(a\ c)(a\ b)$ é um produto minimal de Γ. Note também que $\{a,b,c\} \subset Supp(\psi(a\ c)(a\ b))$ e $\|\psi(a\ c)(a\ b)\| > \|\psi\|$.
2. $\Delta'''(c\ f)$. Trivial.
3. $\Delta'''(b\ e)$. Trivial.
4. $\Delta'''(c\ f)(b\ e)$.
   (a) $e = c \rightsquigarrow \Delta'''(c\ f)(b\ c)$. Temos que $\Delta'''(c\ f)(b\ e)(a\ c)(a\ b) = \Delta'''(a\ c)(a\ f)$. Como b não está em $\Delta'''(a\ c)(a\ f)$, $(a\ b\ c)$ é supérfluo.
   (b) $e \notin \{a,b,c\} \rightsquigarrow \Delta'''(c\ f)(b\ e)$.
      i. $\{b,c\}$ estão no mesmo ciclo α_i. Pelo Lema 2.2.3, $\alpha_i = (c\ f\ldots b\ e\ldots)$. Então temos que

$$\begin{aligned}\alpha_i(a\ c)(a\ b) &= (c\ f\ldots b\ e\ldots)(a\ c)(a\ b)\\ &= (c\ a\ f\ldots b\ e\ldots)(a\ b)\\ &= (c\ b\ f\ldots)(a\ e\ldots) = \beta.\end{aligned}$$

      Observe que $\|\beta\| = \|\alpha_i\|$ e que β pode ser reescrito de forma minimal como $\psi(a\ e)(b\ c)$. Suponha Λ um produto minimal de $\alpha_1\alpha_2\ldots\alpha_{i-1}\alpha_{i+1}\ldots\alpha_m$. O produto $\Lambda\psi(a\ e)(b\ c)$ é um produto minimal de Γ.
      ii. $\{b,c\}$ estão em ciclos diferentes. Lema 4.2.1.
5. $\Delta'''(a\ d)$. Trivial.
6. $\Delta'''(c\ f)(a\ d)$.
   (a) $d = b \rightsquigarrow \Delta'''(c\ f)(a\ b)$. Temos que $\Delta'''(c\ f)(a\ b)(a\ c)(a\ b) = \Delta'''(c\ f)(b\ c)$. Portanto, $(a\ b\ c)$ é supérfluo.
   (b) $d = c \rightsquigarrow \Delta'''(c\ f)(a\ c)$. Temos que $\Delta'''(c\ f)(a\ c)(a\ c)(a\ b) = \Delta'''(c\ f)(a\ b)$. Trivial.
   (c) $d \notin \{a,b,c\} \rightsquigarrow \Delta'''(c\ f)(a\ d)$.
      i. $\{a,c\}$ estão no mesmo ciclo α_i. Pelo Lema 2.2.3, $\alpha_i = (c\ f\ldots a\ d\ldots)$. Então temos que

$$\begin{aligned}\alpha_i(a\ c)(a\ b) &= (c\ f\ldots a\ d\ldots)(a\ c)(a\ b)\\ &= (a\ f\ldots)(c\ d\ldots)(a\ b)\\ &= (a\ b\ f\ldots)(c\ d\ldots) = \beta.\end{aligned}$$

Observe que $\|\beta\| = \|\alpha_i\|$ e que β pode ser reescrito de forma minimal como $\psi(c\ d)(a\ b)$. Suponha Λ um produto minimal de $\alpha_1\alpha_2\ldots\alpha_{i-1}\alpha_{i+1}\ldots\alpha_m$. O produto $\Lambda\psi(c\ d)(a\ b)$ é um produto minimal de Γ.

ii. $\{a, c\}$ estão em ciclos diferentes. Lema 4.2.1.

7. $\Delta'''(b\ e)(a\ d)$.

   (a) $d = b, e = c \rightsquigarrow \Delta'''(b\ c)(a\ b)$. Temos que $\Delta'''(b\ c)(a\ b)(a\ c)(a\ b) = \Delta'''$. Portanto, $(a\ b\ c)$ é supérfluo.

   (b) $d = c, e = c \rightsquigarrow \Delta'''(b\ c)(a\ c)$. Temos que $\Delta'''(b\ c)(a\ c)(a\ c)(a\ b) = \Delta'''(a\ b)(a\ c)$. Trivial.

   (c) $d \notin a, b, c, e = c \rightsquigarrow \Delta'''(b\ c)(a\ d)$. Temos que $\Delta'''(b\ c)(a\ d)(a\ c)(a\ b) = \Delta'''(a\ d)(a\ c)$. Portanto, $(a\ b\ c)$ é supérfluo.

   (d) $d = b, e \notin \{a, b, c\} \rightsquigarrow \Delta'''(b\ e)(a\ b)$. Temos que $\Delta'''(b\ e)(a\ b)(a\ c)(a\ b) = \Delta'''(b\ e)(b\ c)$. Portanto, $(a\ b\ c)$ é supérfluo.

   (e) $d = c, e \notin \{a, b, c\} \rightsquigarrow \Delta'''(b\ e)(a\ c)$. Temos que $\Delta'''(b\ e)(a\ c)(a\ c)(a\ b) = \Delta'''(b\ e)(a\ b)$. Portanto, $(a\ b\ c)$ é supérfluo.

   (f) $d \notin a, b, c, e \notin \{a, b, c\} \rightsquigarrow \Delta'''(b\ e)(a\ d)$

      i. $\{a, b\}$ estão no mesmo ciclo α_i. Pelo Lema 2.2.3, $\alpha_i = (a\ d\ldots b\ e\ldots)$. Então temos que

$$\begin{aligned}\alpha_i(a\ c)(a\ b) &= (a\ d\ldots b\ e\ldots)(a\ c)(a\ b)\\ &= (a\ c\ d\ldots b\ e\ldots)(a\ b)\\ &= (b\ c\ d\ldots)(a\ e\ldots) = \beta.\end{aligned}$$

      Observe que $\|\beta\| = \|\alpha_i\|$ e que β pode ser reescrito de forma minimal como $\psi(a\ e)(b\ c)$. Suponha Λ um produto minimal de $\alpha_1\alpha_2\ldots\alpha_{i-1}\alpha_{i+1}\ldots\alpha_m$. O produto $\Lambda\psi(a\ e)(b\ c)$ é um produto minimal de Γ.

      ii. $\{a, b\}$ estão em ciclos diferentes. Lema 4.2.1.

8. $\Delta'''(c\ f)(b\ e)(a\ d)$.

   (a) $d = b, e = c \rightsquigarrow \Delta'''(c\ f)(b\ c)(a\ b)$. Temos que $\Delta'''(c\ f)(b\ c)(a\ b)(a\ c)(a\ b) = \Delta'''(c\ f)$. Portanto, $(a\ b\ c)$ é supérfluo.

   (b) $d = c, e = c \rightsquigarrow \Delta'''(c\ f)(b\ c)(a\ c)$. Temos que $\Delta'''(c\ f)(b\ c)(a\ c)(a\ c)(a\ b) = \Delta'''(c\ f)$. Portanto, $(a\ b\ c)$ é supérfluo.

   (c) $d \notin a, b, c, e = c \rightsquigarrow \Delta'''(c\ f)(b\ c)(a\ d)$. Temos que $\Delta'''(c\ f)(a\ d)(a\ d)(a\ c)(a\ b) = \Delta'''(c\ f)(a\ d)(a\ c)$. Portanto, $(a\ b\ c)$ é supérfluo.

   (d) $d = b, e \notin \{a, b, c\} \rightsquigarrow \Delta'''(c\ f)(b\ e)(a\ b)$. Temos que $\Delta'''(c\ f)(b\ e)(a\ b)(a\ c)(a\ b) = \Delta'''(c\ f)(b\ e)(b\ c)$. Portanto, $(a\ b\ c)$ é supérfluo.

   (e) $d = c, e \notin \{a, b, c\} \rightsquigarrow \Delta'''(c\ f)(b\ e)(a\ c)$. Temos que $\Delta'''(c\ f)(b\ e)(a\ c)(a\ c)(a\ b) = \Delta'''(b\ e)(c\ f)(a\ b)$ Trivial.

   (f) $d \notin \{a, b, c\}$, $e \neq (a, b, c) \rightsquigarrow \Delta'''(c\ f)(b\ e)(a\ d)$.

i. $\{a, b\}$ estão no mesmo ciclo α_i e $\{c\}$ em α_j. Pelo Lema 2.2.3, $\alpha_i = (a\ d \dots b\ e \dots)$ e $\alpha_j = (c\ f \dots)$. Então temos que

$$\begin{aligned}\alpha_i\alpha_j(a\ c)(a\ b) &= (a\ d \dots b\ e \dots)(c\ f \dots)(a\ c)(a\ b)\\ &= (a\ f \dots c\ d \dots b\ e \dots)(a\ b)\\ &= (b\ f \dots c\ d \dots)(a\ e \dots) = \beta.\end{aligned}$$

Observe que $\|\beta\| = \|\alpha_i\alpha_j\|$ e que β pode ser reescrito de forma minimal como $\psi(a\ e)(b\ c)$.
Suponha Λ um produto minimal de $\alpha_1\alpha_2 \dots \alpha_{i-1}\alpha_{i+1} \dots \alpha_{j-1}\alpha_{j+1} \dots \alpha_m$. O produto $\Lambda\psi(a\ e)(b\ c)$ é um produto minimal de Γ.

ii. $\{a, c\}$ estão no mesmo ciclo α_i e $\{b\}$ em α_j. Pelo Lema 2.2.3, $\alpha_i = (a\ d \dots c\ f \dots)$ e $\alpha_j = (b\ e \dots)$. Então temos que

$$\begin{aligned}\alpha_i\alpha_j(a\ c)(a\ b) &= (a\ d \dots c\ f \dots)(b\ e \dots)(a\ c)(a\ b)\\ &= (c\ d \dots)(a\ f \dots)(b\ e \dots)(a\ b)\\ &= (b\ f \dots a\ e \dots)(c\ d \dots) = \beta.\end{aligned}$$

Observe que $\|\beta\| = \|\alpha_i\alpha_j\|$ e que β pode ser reescrito de forma minimal como $\psi(c\ d)(a\ b)$.
Suponha Λ um produto minimal de $\alpha_1\alpha_2 \dots \alpha_{i-1}\alpha_{i+1} \dots \alpha_{j-1}\alpha_{j+1} \dots \alpha_m$. O produto $\Lambda\psi(c\ d)(a\ b)$ é um produto minimal de Γ.

iii. $\{b, c\}$ estão no mesmo ciclo α_i e $\{a\}$ em α_j. Pelo Lema 2.2.3, $\alpha_i = (c\ f \dots b\ e \dots)$ e $\alpha_j = (a\ d \dots)$. Então temos que

$$\begin{aligned}\alpha_i\alpha_j(a\ c)(a\ b) &= (c\ f \dots b\ e \dots)(a\ d \dots)(a\ c)(a\ b)\\ &= (a\ f \dots b\ e \dots c\ d \dots)(a\ b)\\ &= (b\ f \dots)(a\ e \dots c\ d \dots) = \beta.\end{aligned}$$

Observe que $\|\beta\| = \|\alpha_i\alpha_j\|$ e que β pode ser reescrito de forma minimal como $\psi(b\ f)(a\ c)$.
Suponha Λ um produto minimal de $\alpha_1\alpha_2 \dots \alpha_{i-1}\alpha_{i+1} \dots \alpha_{j-1}\alpha_{j+1} \dots \alpha_m$. O produto $\Lambda\psi(b\ f)(a\ c)$ é um produto minimal de Γ.

iv. $\{a, b, c\}$ estão todos no mesmo ciclo α_i.

A. Pelo Lema 2.2.3, $\alpha_i = (a\ d \dots c\ f \dots b\ e \dots)$. Então temos que

$$\begin{aligned}\alpha_i(a\ c)(a\ b) &= (c\ f \dots b\ e \dots a\ d \dots)(a\ c)(a\ b)\\ &= (a\ f \dots b\ e \dots)(c\ d \dots)(a\ b)\\ &= (b\ f \dots)(a\ e \dots)(c\ d \dots) = \beta.\end{aligned}$$

Observe que $\|\beta\| < \|\alpha_i\|$. Portanto, nesse caso, $(a\ b\ c)$ é supérfluo.

B. Pelo Lema 2.2.3, $\alpha_i = (a\ d \dots b\ e \dots c\ f \dots)$. Então temos que

$$\begin{aligned}\alpha_i(a\ c)(a\ b) &= (a\ d \dots b\ e \dots c\ f \dots)(a\ c)(a\ b)\\ &= (a\ f \dots)(c\ d \dots b\ e \dots)(a\ b)\\ &= (a\ e \dots c\ d \dots b\ f \dots) = \beta.\end{aligned}$$

Observe que $\|\beta\| = \|\alpha_i\|$ e que β pode ser reescrito de forma minimal como $\psi(a\ b)(a\ c)$.
Suponha Λ um produto minimal de $\alpha_1\alpha_2 \dots \alpha_{i-1}\alpha_{i+1} \dots \alpha_m$. O produto $\Lambda\psi(a\ b)(a\ c)$ é um produto minimal de Γ.

v. $\{a, b, c\}$ estão cada um no suporte de um ciclo diferente, a_i, a_j e a_k, respectivamente. Pelo Teorema 2.2.1, o grafo de $\Delta'''(c\ f)(b\ e)(a\ d)$ é uma floresta, com ao menos três árvores, cada uma referente aos ciclos a_i, a_j e a_k. O efeito da multiplicação de $\Delta'''(c\ f)(b\ e)(a\ d)$ por $(a\ c)(a\ b)$ é o mesmo de conectar as árvores referentes aos ciclos a_i e a_k com uma aresta, gerando um nova árvore, e em seguida, conectar com uma segunda aresta, essa nova árvore com a árvore referente ao ciclo a_j. No fim, teremos que $\Delta'''(c\ f)(b\ e)(a\ d)(a\ c)(a\ b)$ também é uma floresta, e portanto, um produto minimal de Γ.

□

Corolário 4.2.1. Seja Γ um produto da forma $\Delta(a\ b\ c)$, com $(a\ b\ c)$ não-supérfluo. Se Γ é um k-ciclo, então sempre existe um produto minimal Ω de Γ que assume uma das seguintes formas:

1. $\Theta(a\ c)(a\ b)$ ou
2. $\Theta(a\ b)(a\ c)$.

Demonstração. Na demonstração do Lema 4.2.2, repare que em todos os casos em que o produto minimal produzido, correspondendo às propriedades enunciadas, eram produtos cujos dois fatores 2-ciclos mais à direita eram disjuntos e os símbolos movidos por esses fatores estavam em suportes de ciclos diferentes e portanto, não se relacionam com este caso (em que Γ é um k-ciclo). Nos outros casos, os dois fatores 2-ciclos mais à direita eram $(a\ c)(a\ b)$ ou $(a\ b)(a\ c)$. □

Corolário 4.2.2. Seja Γ um produto da forma $\Delta(a\ b\ c)$, com $(a\ b\ c)$ não-supérfluo. Se Γ é um k-ciclo, então sempre existe um produto minimal Ω de Γ que assume uma das seguintes formas:

1. $\Theta(a\ c)(a\ b)$ ou
2. $\Theta(a\ b)(a\ c)$.

Demonstração. Na demonstração do Lema 4.2.2, repare que em todos os casos em que o produto minimal produzido, correspondendo às propriedades enunciadas, eram produtos cujos dois fatores 2-ciclos mais à direita eram disjuntos e os símbolos movidos por esses fatores estavam em suportes de ciclos diferentes e portanto, não se relacionam com este caso (em que Γ é um k-ciclo). Nos outros casos, os dois fatores 2-ciclos mais à direita eram $(a\ c)(a\ b)$ ou $(a\ b)(a\ c)$. □

No Lema 4.2.2, mostramos a relação entre um 3-ciclo não-supérfluo $(a\ b\ c)$ em um produto arbitrário $\Delta(a\ b\ c)$ e seus produtos minimais. Não levamos em consideração se este 3-ciclo $(a\ b\ c)$ ou mesmo o produto $\Delta(a\ b\ c)$ representavam transposições sobre um cromossomo arbitrário. Trataremos este caso nos dois lemas subsequentes.

Lema 4.2.3. Sejam π e σ dois cromossomos sobre E. Se $\sigma\pi^{-1} = \rho_r \dots \rho_2\rho_1$ é um k-ciclo, onde $k \leq n$, e $\rho_1 = (a\ b\ c)$ é não-supérfluo em $\rho_r \dots \rho_2\rho_1$, então existe um produto minimal Γ de $\sigma\pi^{-1}$, tal que $\Gamma = \Delta(a\ c)(a\ b)$.

Demonstração. Pelo Lema 4.2.2, sabemos que se Γ é um produto minimal de $\sigma\pi^{-1}$, então esse produto assume uma das seguintes formas:

$$\Gamma = \begin{cases} \Theta(a\ c)(a\ b) \\ \Theta(a\ b)(a\ c) \\ \Theta(b\ d)(a\ c) \\ \Theta(c\ d)(a\ b) \\ \Theta(a\ d)(b\ c). \end{cases}$$

O produto Γ pode assumir apenas as formas $\Theta(a\ c)(a\ b)$ ou $\Theta(a\ b)(a\ c)$, pois pelo Corolário 4.2.2, se Γ assumisse qualquer uma das outras formas, $\sigma\pi^{-1}$ não seria um k-ciclo e portanto teríamos uma contradição. Por outro lado, Γ poderia ser $\Theta(a\ b)(a\ c)$. O problema é que o Lema 4.2.2 não leva em conta se Γ é resultante de um produto de dois n-ciclos. Se levarmos este fato em consideração, pela Proposição 4.1.1, o caso em que Γ seria equivalente a um produto $\Theta(a\ b)(a\ c)$ é descartado. Portanto, Γ só pode ser da forma $\Theta(a\ c)(a\ b)$. □

Lema 4.2.4. Sejam π e σ dois cromossomos sobre E. Se $\sigma\pi^{-1} = \rho_r \dots \rho_2\rho_1$ não é uma permutação cíclica e $\rho_1 = (a\ b\ c)$ é não-supérfluo em $\rho_r \dots \rho_2\rho_1$, então:

(1) existe um produto minimal Γ de $\sigma\pi^{-1}$ tal que $\Gamma = \Delta(a\ c)(a\ b)$; ou

(2) existe um produto minimal Γ de $\sigma\pi^{-1}$ tal que $\Gamma = \Delta(c\ d)(a\ b)$.

Demonstração.

(1) Seja $C_s \dots C_1$ a decomposição em ciclos disjunta de $\sigma\pi^{-1}$. Suponha que algum ciclo C_i tenha como produto minimal $\Sigma = \Pi(a\ c)(a\ b)$. Como os ciclos $C_1, \dots, C_s$ comutam entre si, podemos escrever $\sigma\pi^{-1}$ como $C_s \dots C_{i+1}C_{i-1} \dots C_1C_i$. Seja Λ um produto minimal de $C_s \dots C_{i+1}C_{i-1} \dots C_1$. Como Λ e Σ são disjuntos e não contêm a, b ou c, então pelo Lema 2.2.2, $\Lambda\Delta(a\ c)(a\ b)$ é um produto minimal de $\sigma\pi^{-1}$.

(2) O Lema 4.2.2 afirma que existe um produto minimal de $\sigma\pi^{-1}$ cujos dois fatores mais à direita contêm a, b e c. O caso em que este produto minimal termina com $(a\ c)(a\ b)$ já foi tratado no item (1). Então, um produto minimal de $\sigma\pi^{-1}$ só pode ser da forma $\Delta(b\ d)(a\ c)$, $\Delta(c\ d)(a\ b)$ ou $\Delta(a\ d)(b\ c)$. Sem perda de generalidade, considere todos os casos como casos em que os dois 2-ciclos são disjuntos, do tipo $(c\ d)(a\ b)$. Neste caso, sendo $\sigma\pi^{-1}$ uma permutação par, pelo Lema 2.1.2, os dois 2-ciclos mais à direita das formas mencionadas correspondem aos geradores de ρ_1 e ρ_2. Portanto, $\rho_1 = (a\ b\ c)$ e $\rho_2 = (a\ d\ c)$.

□

Se derivarmos limites para o problema da distância de transposição a partir dos dois últimos lemas, obteremos os mesmos limites da seção anterior.

4.3 Algoritmo de aproximação com razão 2

A ideia básica do algoritmo consiste em procurar pelos 3-ciclos previstos nos Lemas 4.1.1 e 4.1.3.

O algoritmo começa computando $\sigma\pi^{-1}$. Em seguida, testa se a permutação resultante é um k-ciclo. Caso positivo, o algoritmo procura por uma tripla (a, b, c) tal que o 3-ciclo $\rho = (a\ b\ c)$ seja aplicável à π, de acordo com o Lema 4.1.1. Esta tripla deve ser tal que ambos $\sigma\pi^{-1}$ e π sejam da forma $(a \dots b \dots c \dots)$. Se o algoritmo for implementado de modo a utilizar a heurística do Lema 4.1.2, então, antes dessa busca, o algoritmo deve buscar uma tripla (a, b, c) tal que $\sigma\pi^{-1} = (a\ b\ c \dots)$ e $\pi = (a \dots b \dots c \dots)$.

Por outro lado, se $\sigma\pi^{-1}$ não resultar em um k-ciclo, então para cada ciclo C_i da decomposição disjunta de $\sigma\pi^{-1}$, o algoritmo procurará por uma tripla (a, b, c) tal que ambos C_i e π sejam da forma $(a \dots b \dots c \dots)$. Esta etapa corresponde à parte (1) do Lema 4.1.3. Se mais de uma tripla for encontrada, o algoritmo escolhe a que está em um ciclo de comprimento ímpar. Se tal tripla não for encontrada, então para cada par de ciclos C_i e C_j, o algoritmo procurará por uma quádrupla (a, b, c, d) tal que $\{a, b\} \in Supp(C_i)$ e $\{c, d\} \in Supp(C_j)$ tais que $(a\ b\ c)$ é aplicável a π e $(a\ d\ c)$ é aplicável à $(a\ b\ c)\pi$, i.e., π é da forma $(a \dots d \dots b \dots c \dots)$. Esta etapa corresponde à parte (2) do Lema 4.1.3.

Algoritmo 3 Algoritmo algébrico de ordenação por transposições com razão de aproximação 2

Entrada: duas permutações π e σ
Saída: lista L contendo uma sequência de transposições capaz de transformar π em σ

1: crie uma lista vazia L
2: $\mu \leftarrow \sigma\pi^{-1}$
3: **enquanto** $\mu \neq \iota$ **faça**
4: **se** μ é um k-ciclo **então**
5: procure $\rho = (a\ b\ c)$ ▷ Lema 4.1.2
6: **se** existe ρ **então**
7: $\pi \leftarrow \rho\pi$
8: adicione ρ no final da lista L ▷ o segundo 3-ciclo aplicável, previsto pelo Lema 4.1.2, será descoberto na próxima iteração.
9: **senão**
10: encontre $\rho = (a\ b\ c)$ ▷ Lema 4.1.1
11: $\pi \leftarrow \rho\pi$
12: adicione ρ no final da lista L
13: **fim se**
14: **senão**
15: procure $\rho = (a\ b\ c)$ ▷ parte (1) do Lema 4.1.3
16: **se** existe ρ **então**
17: $\pi \leftarrow \rho\pi$
18: adicione ρ no final da lista L
19: **senão**
20: encontre $\rho' = (a\ b\ c)$ e $\rho'' = (a\ d\ c)$ ▷ parte (2) do Lema 4.1.3
21: $\pi \leftarrow \rho''\rho'\pi$
22: adicione ρ' no final da lista L
23: adicione ρ'' no final da lista L
24: **fim se**
25: **fim se**
26: $\mu \leftarrow \sigma\pi^{-1}$
27: **fim enquanto**
28: **retorne** L

4.4 Análise de complexidade

Como o foco principal deste trabalho foi a investigação de um método para resolver o problema da distância de transposição desvinculado do formalismo de Bafna e Pevzner [4], nesta seção faremos uma análise do tempo de pior caso do algoritmo proposto na seção anterior, suficiente apenas para demonstrar que ele executa em tempo polinomial.

Teorema 4.4.1. O Algoritmo 3 executa em $O(n^5)$.

Demonstração. Para tornar mais simples a análise de complexidade, dividiremos o Algoritmo 4.3 em sub-rotinas.

1. Rotina que toma um n-ciclo π e monta uma tabela *hash* com todas triplas (a, b, c) tais que $\pi = (a \dots b \dots c \dots)$. Esta rotina pode ser implementada de modo a executar em $O(n^3)$. Sua chamada é feita logo depois de se entrar no laço mais externo do algoritmo, na linha 3. O propósito dessa tabela *hash* é permitir o teste de aplicabilidade de um 3-ciclo em π em tempo constante $O(1)$.

2. Rotina que toma um n-ciclo π e monta uma tabela *hash* com todas as quádruplas (a, b, c, d) tais que $\pi = (a \dots d \dots b \dots c \dots)$. Esta rotina pode ser implementada de modo a executar em $O(n^4)$. Sua chamada é feita logo depois da chamada da rotina anterior. Essa tabela *hash* permitirá o teste de aplicabilidade de dois 3-ciclos em π em tempo constante $O(1)$.

3. Rotina que toma um k-ciclo μ e uma tabela *hash* e verifica se existe alguma tripla (a, b, c) tal que $\mu = (a\ b\ c \dots)$ e esta tripla está na tabela *hash* fornecida. Esta rotina executa em tempo linear $O(n)$ e corresponde à chamada na linha 4.

4. Rotina que toma um k-ciclo μ e uma tabela *hash* e verifica se existe alguma tripla (a, b, c) tal que $\mu = (a \dots b \dots c \dots)$ e esta tripla está na tabela *hash* fornecida. Esta rotina executa em $O(n^3)$ e corresponde à chamada na linha 9.

5. Rotina que toma uma permutação μ e uma tabela *hash* e verifica se existe alguma tripla (a, b, c) tal que algum ciclo C_i da decomposição disjunta de μ é da forma $(a \dots b \dots c \dots)$ e esta tripla está na tabela *hash* fornecida. Esta rotina executa em $O(n^3)$ e corresponde à chamada na linha 14.

6. Rotina que toma uma permutação μ e uma tabela *hash* e verifica se existe alguma quádrupla (a, b, c, d) tal que existem ciclos C_i e C_j na decomposição disjunta de μ tais que $C_i = (a \dots b \dots)$ e $C_j = (c \dots d \dots)$ e a quádrupla está na tabela *hash* fornecida. Esta rotina executa em $O(n^4)$ e corresponde à chamada na linha 19.

Como o laço mais externo do algoritmo, que controla a parada quando $\sigma\pi^{-1}$ se igualar à ι, tem complexidade linear, pois $\|\sigma\pi^{-1}\| \leq n$ e este valor diminui em 2 a cada iteração, a complexidade do Algoritmo 4.3 é $O(n^5)$. □

Na implementação, todas as operações envolvendo permutações, como a aplicação de transposições e o cálculo de $\sigma\pi^{-1}$, foram delegadas ao GAP [32]. Procuramos na documentação deste sistema informações a respeito de complexidade computacional, mas não encontramos. De toda forma, supomos que a complexidade de tempo dessas operações sejam no máximo lineares. Como essas operações ocorrem fora dos procedimentos de busca

das triplas e quádruplas mencionadas, acabam por não influenciar na determinação da complexidade de tempo de pior caso geral do algoritmo.

A complexidade de espaço do algoritmo, por sua vez, é $O(n^4)$ e é determinada pela criação da tabela *hash* relacionada ao item 2.

Capítulo 5

Implementação

Neste capítulo, apresentaremos detalhes da implementação do Algoritmo 4.3, na Seção 5.1. Em seguida, na Seção 5.2, discutiremos os resultados obtidos em relação à auditoria feita na ferramenta GRAAu [30].

5.1 Implementação do algoritmo

Foi desenvolvido um aplicativo *desktop*, baseado em Java Swing, implementando o Algoritmo 4.3 proposto no capítulo anterior. Neste aplicativo, apenas as estruturas de controle do algoritmo foram implementadas em Java. As operações envolvendo permutações foram delegadas ao GAP [32], um pacote de álgebra computacional.

O aplicativo consiste de um formulário onde o usuário fornece valores para os n-ciclos π e σ, representando, respectivamente, um cromossomo inicial e um final. Se o usuário não fornecer nenhuma informação para σ, a permutação que representa o cromossomo $(0, 1, 2 \dots n-1)$ é assumida. Para calcular a sequência de transposições que transformará π em σ, o usuário então pressiona o botão *Transfom.* Em seguida, o sistema calcula as transposições necessárias para a transformação e imprime na área rotulada como *Output* as informações acerca de cada iteração correspondente à execução do algoritmo.

O código-fonte do aplicativo está disponível em um sítio[1] na plataforma *Google Code*. Neste sítio, encontra-se um roteiro mostrando os passos a serem seguidos para compilar e executar o aplicativo.

5.2 Auditoria

A auditoria de um algoritmo de rearranjo de genomas consiste na geração das distâncias de rearranjo, para todas as permutações de até um certo tamanho, pelo algoritmo auditado e a comparação dessas distâncias com os valores exatos. A auditoria é útil para medir a qualidade das respostas de um algoritmo de rearranjo.

A ferramenta GRAAu [30], acrônimo para *Genome Rearrangement Algorithm Auditor*, é uma ferramenta de auditoria de algoritmos de rearranjos (uma ampla gama de tipos de rearranjos é suportada pela ferramenta) capaz de auditar as respostas para todas as permutações de até tamanho 13, para permutações sem sinais, e 10, para permutações

[1] `http://code.google.com/p/algebraic-sort-by-transp/m`

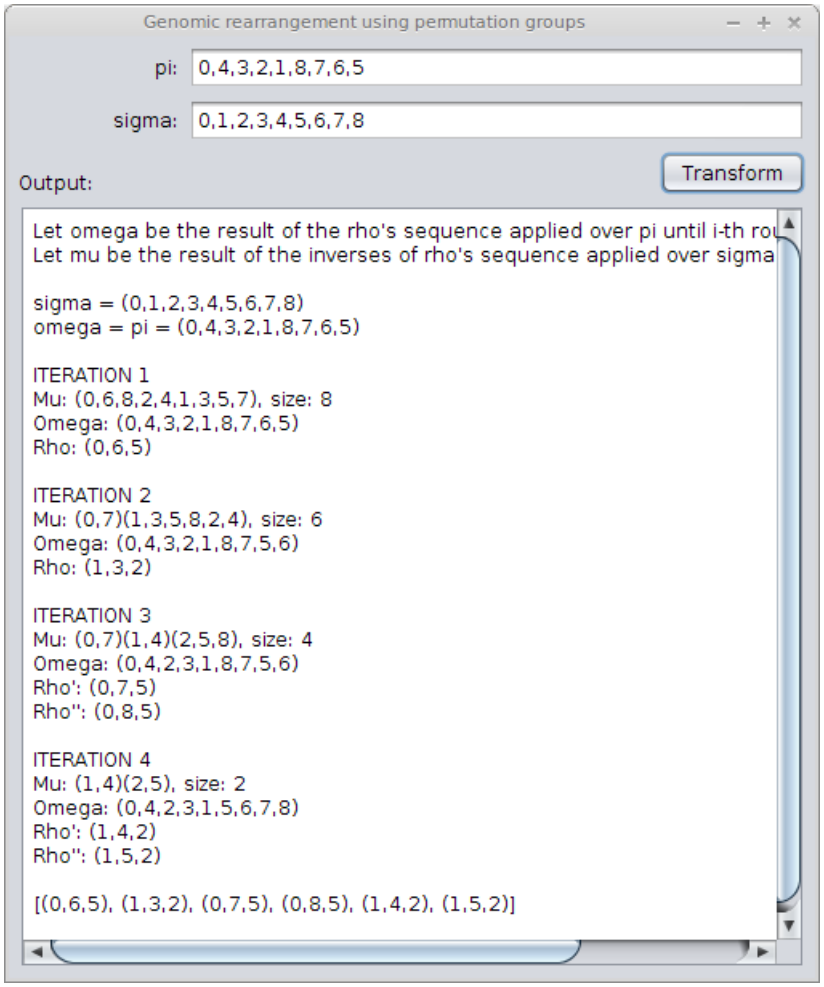

Figura 5.1: Captura de tela do aplicativo implementado, mostrando a ordenação do cromossomo $(0, 4, 3, 2, 1, 8, 7, 6, 5)$.

com sinais. Tanto quanto sabemos, não há outra ferramenta conhecida na literatura de Rearranjo de Genomas com tais capacidades.

O GRAAu consiste de um banco de dados, chamado *Rearrangement Distance Database* [31], que contém armazenadas as distâncias de rearranjo exatas calculadas para todas as permutações dos tamanhos mencionados, associado a um serviço *web*, pelo qual as distâncias de rearranjo calculadas pelo algoritmo auditado são enviadas para um servidor, que então calcula diversas métricas de performance baseadas nessas respostas. Ao longo e ao fim do processo de auditoria, todas as métricas calculadas ficam disponíveis para consulta na *web*[2].

As métricas calculadas e disponibilizadas pela ferramenta GRAAu são:

- **Diâmetro**: valor da maior distância de rearranjo entre duas permutações quaisquer, calculada e retornada pelo algoritmo auditado;
- **Distância média**: média aritmética das distâncias retornadas pelo algoritmo auditado;

[2] `http://mirza.ic.unicamp.br:8080/bioinfo/index.jsf`

- **Razão média**: média aritmética das razões entre a distância calculada pelo algoritmo auditado e a distância exata registrada no banco de dados;
- **Razão máxima**: razão máxima dentre as razões determinadas pela distância calculada pelo algoritmo auditado e a distância exata registrada no banco de dados;
- **Igualdade**: percentual de permutações para as quais o algoritmo auditado respondeu o mesmo valor da distância exata registrada no banco de dados.

O algoritmo implementado na seção anterior foi auditado na ferramenta GRAAu em duas versões: com e sem a heurística do Lema 4.1.2. Os resultados dessa auditoria, para permutações de tamanho até 11, estão disponíveis nas Tabelas 5.1 e 5.2. Chamamos o algoritmo implementado sem a heurística de "LMN", e de "LMNh", a versão com a heurística. Repare que com a heurística, as métricas de performance do algoritmo ficam ligeiramente melhores para as permutações de tamanho $n \geq 6$.

Tabela 5.1: Resultado da auditoria do algoritmo LMN na ferramenta GRAAu.

n	**Diâmetro**	**Distância média**	**Razão média**	**Razão máxima**	**Igualdade**
1	0	0	1	1	100%
2	1	0.5	1	1	100%
3	2	1	1	1	100%
4	3	1.54	1	1	100%
5	4	2.08	1	1.3333334	99.17%
6	5	2.64	1.01	1.3333334	96.11%
7	6	3.2	1.02	1.6666666	93.23%
8	7	3.76	1.03	1.6666666	88.41%
9	8	4.32	1.04	1.75	84.36%
10	9	4.88	1.05	1.75	78.71%
11	10	5.44	1.06	1.8	74.17%

Na Tabela 5.3, o percentual de igualdade obtido por nosso algoritmo é comparado a outros resultados disponíveis na literatura. Podemos observar que a razão de aproximação não está diretamente relacionada ao percentual de igualdade. Isso fica evidente quando notamos que o algoritmo WDM, embora com razão de aproximação 2.25, apresenta resultados melhores do que os produzidos pelo algoritmo proposto neste trabalho — com razão 2 — e melhores até que os resultados do algoritmo de Hartman, com razão 1.5.

Tabela 5.2: Resultado da auditoria do algoritmo LMNh na ferramenta GRAAu.

n	Diâmetro	Distância média	Razão média	Razão máxima	Igualdade
1	0	0	1	1	100%
2	1	0.5	1	1	100%
3	2	1	1	1	100%
4	3	1.54	1	1	100%
5	4	2.08	1	1.3333334	99.17%
6	5	2.63	1.01	1.3333334	97.08%
7	6	3.18	1.02	1.6666666	94.35%
8	7	3.73	1.03	1.6666666	91.06%
9	8	4.29	1.03	1.75	86.91%
10	9	4.83	1.04	1.75	82.99%
11	10	5.4	1.05	1.8	77.99%

Tabela 5.3: Comparação dos resultados de igualdade obtidos pelo algoritmo proposto neste trabalho com outros disponíveis na literatura. Os rótulos das colunas da tabela são: WDM - Walter, Dias e Meidanis [66] - razão 2.25, Hartman [36] - razão 1.5 (implementado por Honda [40]), BP - Bafna e Pevzner [6] - razão 1.5 (implementado por Oliveira [56]), LMN (este trabalho) e LMNh (este trabalho)

n	Número de permutações	WDM	Hartman	BP	LMN	LMNh
1	1	100%	100%	100%	100%	100%
2	2	100%	100%	100%	100%	100%
3	6	100%	100%	100%	100%	100%
4	24	100%	100%	100%	100%	100%
5	120	100%	100%	100%	99.17%	99.17%
6	720	99.16%	99.72%	100%	96.11%	97.08%
7	5040	98.57%	97.85%	99.98%	93.23%	94.35%
8	40320	97.12%	96.23%	99.66%	88.41%	91.06%
9	362880	96.06%	92.99%	98.79%	84.36%	86.91%

Capítulo 6

Conclusões

Nesta dissertação, propomos um algoritmo de aproximação de razão 2 para o problema da distância de transposição, com base em resultados da Teoria de Grupos de Permutações. Embora a razão não seja a melhor conhecida, a contribuição desta dissertação é teórica, pois propõe um algoritmo usando apenas Álgebra, de forma desvinculada do algoritmo de Bafna e Pevzner [6]. É importante notar que nossa solução formaliza, de maneira natural, o método de Bafna e Pevzner, e traz provas para certos fatos não demonstrados por eles.

Na investigação de uma abordagem alternativa ao problema, encontramos o conceito de produtos minimais de 2-ciclos e o algoritmo de redução de permutações proposto por Higgs [39]. Esses dois resultados mostraram-se bastante interessantes, pois a geração dos casos e a análise de boa parte deles poderá ser automatizada, permitindo diminuir a razão de aproximação e possivelmente obter uma nova prova de que o problema da distância de transposição é $\mathcal{NP}$-difícil.

Durante o levantamento bibliográfico feito neste trabalho, notamos que a definição de 2-norma de uma permutação, introduzido no formalismo algébrico, é equivalente ao número de fatores em um produto minimal de 2-ciclos. Constatamos que alguns resultados referentes à 2-norma já tinham sido demonstrados de forma equivalente em trabalhos anteriores.

Um resultado interessante deste trabalho diz respeito a inexistência de grafos-de-ciclos com apenas um ciclo não-orientado ou constituídos de apenas ciclos não-orientados que não se cruzam no formalismo clássico de Bafna e Pevzner [6] (considerando a equivalência entre $G(\pi)$ e $\sigma\pi^{-1}$, conforme mostrado na Seção 3.2.2). Demonstramos formalmente que tais grafos-de-ciclos não existem (veja proposições 4.1.1 e 4.1.2). Além disso, demonstramos algebricamente um limite superior para o problema da distância de transposição, que é equivalente ao estabelecido por Bafna e Pevzner no Teorema 3.1.4. Este limite não tinha sido demonstrado no formalismo algébrico de Dias e Meidanis [49].

Voltando aos produtos minimais de 2-ciclos, nossa expectativa inicial era de obter todas as demonstrações de existência dos 3-ciclos aplicáveis e dos limites de aproximação utilizando apenas esses resultados. Porém, ao longo do trabalho, notamos que para apresentar uma solução utilizando apenas resultados ligados aos produtos minimais de 2-ciclos, necessitaríamos de uma hipótese: a de 3-ciclos não supérfluos (Seção 4.2). O problema é que, ao impedir que 3-ciclos representando transposições sejam cancelados em uma sequência de 3-ciclos transformando um cromossomo inicial em um final, implicitamente estávamos conjecturado um limite superior. Porém, usando fatorações de n-ciclos,

demonstramos formalmente este limite, embora de maneira não construtiva, de forma diferente do que pretendíamos inicialmente. Não obstante, a demonstração mostrou que a intuição para a definição de 3-ciclos supérfluos estava correta.

Como trabalhos futuros, pensamos que a melhor abordagem será usar os argumentos não construtivos para estabelecer novos limites de aproximação, especialmente do limite superior (justificando a hipótese de 3-ciclos não-supérfluos), e utilizar os resultados de produtos minimais de 2-ciclos e redução de permutações para diminuir a razão de aproximação e para obter uma nova prova de que o problema da distância de transposição é $\mathcal{NP}$-difícil.

Referências

[1] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, 25(17):3389–402, 1997. 1

[2] D. A. S. Anjos, G. G. Zerlotini, G. A. Pinto, G. P. Telles, C. J. M. Viana, N. A. Franco, M. E. M. T. Walter, and M. M. Brigido. A method for inferring biological functions using homologous genes among three genomes. In Springer, editor, *Advances in Bioinformatics and Computational Biology - Lecture Notes in Binformatics from the Brazilian Symposium on Bioinformatics - BSB 2007*, volume 4643, pages 69–80, 2007. 1

[3] D. A. Bader, B. M. E. Moret, and M. Yan. A linear-time algorithm for computing inversion distance between signed permutations with an experimental study. In *7th International Workshop on Algorithms and Data Structures*, pages 2365–376, 2001. 1, 7, 8

[4] V. Bafna and P. A. Pevzner. Sorting by transpositions. In *Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 614–623, San Francisco, USA, 1995. 7, 46

[5] V. Bafna and P. A. Pevzner. Genome rearrangements and sorting by reversals. *SIAM J. Comput*, 25(2):272–289, 1996. 1, 7

[6] V. Bafna and P. A. Pevzner. Sorting by transpositions. *SIAM Journal on Discrete Mathematics*, 11(2):224–240, 1998. vii, 1, 10, 19, 26, 27, 29, 31, 51, 52

[7] A. Bergeron, J. Mixtacki, and J. Stoye. A unified view of genome rearrangements. In Springer, editor, *Lecture Notes in Computer Science - Algorithms in Bioinformatics*, volume 4175, pages 163–173, Berlin, Heidelberg, 2006. 9

[8] G. Berkolaiko, J. M. Harrison, and M. Novaes. On inequivalent factorizations of a cycle. *arXiv preprint arXiv:0809.3476*, 2008. 17

[9] P. Berman, S. Hannenhalli, and M. Karpinski. 1.375-approximation algorithm for sorting by reversals. In Springer, editor, *Lecture Notes in Computer Science - Proceedings of 10th European Symposium on Algorithms (ESA'02)*, volume 2461, pages 200–210, Berlin - Heidelberg, 2002. 8

[10] P. Berman, S. Hannenhalli, and M. Karpinski. *1.375-approximation algorithm for sorting by reversals*. Springer, 2002. 7

[11] P. Biane. Minimal factorizations of a cycle and central multiplicative functions on the infinite symmetric group. *Journal of Combinatorial Theory, Series A*, 76(2):197 – 212, 1996. 16

[12] L. Bulteau, G. Fertin, and I. Rusu. Sorting by Transpositions is Difficult. *CoRR*, abs/1011.1157:1–37, 2010. http://arxiv.org/abs/1011.1157v1. 8

[13] A. Caprara. Sorting by reversals is difficult. In ACM Press, editor, *Proceedings of the First Annual International Conference on Computational Molecular Biology (RECOMB'97)*, pages 75–83, New York, NY, USA, 1997. 7

[14] D. A. Christie. Sorting permutations by block-interchanges. *Information Processing Letters*, 60:165–169, 1996. 8, 9

[15] D. A. Christie. A 3/2-approximation algorithm for sorting by reversals. In *Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms*, pages 244–252. Society for Industrial and Applied Mathematics, 1998. 7

[16] D. A. Christie. *Genome Rearrangement Problems*. PhD thesis, Glasgow University, 1998. 8

[17] C. Darwin. *A origem das espécies*. Editora Itatiaia, 2002. 1

[18] Z. Dias and J. Meidanis. Genome rearrangements distance by fusion, fission, and transposition is easy. In *Proceedings of the String Processing and Information Retrieval: 8th International Symposium (SPIRE 2001)*, pages 250–253, 2001. 9

[19] Z. Dias and J. Meidanis. Sorting by prefix transpositions. In Springer, editor, *Lecture Notes in Computer Science - Proceedings of the String Processing and Information Retrieval: 9th International Symposium (SPIRE 2002)*, volume 2476, pages 463–468, Berlin, Heidelberg, 2002. 8, 9

[20] J. Dénes. The representation of a permutation as the product of a minimal number of transpositions, and its connection with the theory of graphs. *Publ. Math. Inst. Hung. Acad. Sci.*, 4:63–71, 1959. 15, 16

[21] T. Dobzhansky and A. H. Sturtevant. Inversions in the chromosomes of *drosophila pseudoobscura*. *Genetics*, 23:28–64, 1938. 1

[22] J. A. Eidswick. Short factorizations of permutations into transpositions. *Discrete Mathematics*, 73(3):239–243, 1989. 16

[23] I. Elias and T. Hartman. A 1.375-approximation algorithm for sorting by transpositions. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(4):369–379, 2006. 8, 19

[24] P. Feijao and J. Meidanis. Scj: a variant of breakpoint distance for which sorting, genome median and genome halving problems are easy. In *Proceedings of the 9th international conference on Algorithms in bioinformatics - WABI'09*, pages 85–96, Heidelberg, 2009. Springer-Verlag. 1, 7, 8, 9

[25] P. C. Feijão and J. Meidanis. A Survey on Genome Rearrangement Problems and Gene Order Based Phylogenies. Technical Report IC-08-033, Instituto de Computação, UNICAMP, Dezembro 2008. vii, 8

[26] M. S. Soares Felipe, R. V. Andrade, N. F. Almeida, M. J. A. Carvalho, Rede Genoma Centro-Oeste, M. E. M. T. Walter, and M. M. Brígido. Transcriptional profiles of the human pathogenic fungus paracoccidioides brasiliensis in mycelium and yeast cells. *The Journal of Biological Chemistry*, 280(1074):24706–24714, 2005. 1

[27] J. Feng and D. Zhu. Faster algorithms for sorting by transpositions and sorting by block interchanges. *ACM Transactions on Algorithms (TALG)*, 3(3):25, 2007. 9

[28] G. Fertin, A. Labarre, I. Rusu, E. Tannier, and S. Vialette. *Combinatorics of Genome Rearrangements*. The MIT Press, 1st edition, 2009. vi, 2, 4, 7

[29] J. Gallian. *Contemporary Abstract Algebra*. Brooks Cole, 7 edition, January 2009. 11

[30] G. R. Galvão and Z. Dias. Graau: Genome rearrangement algorithm auditor. In *In Proceedings of the 4th International Conference on Bioinformatics and Computational Biology (BICoB'2012)*, pages 97–101, 2012. 10, 35, 48

[31] G. R. Galvão and Z. Dias. Computing rearrangement distance of every permutation in the symmetric group. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, pages 106–107. ACM, 2011. 49

[32] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.6.5*, 2013. 10, 46, 48

[33] A. Garcia and Y. Lequain. *Elementos de Alegbra (in Portuguese)*. IMPA, Rio de Janeiro, Brazil, 2009. 11

[34] D. A. Gewurz and F. Merola. Some factorisations counted by catalan numbers. *European Journal of Combinatorics*, 27(6):990 – 994, 2006. 16

[35] S. Hannenhalli and P. A. Pevzner. Transforming cabbage into turnip: polynomial algorithm for sorting signed permutations by reversals. In *Annual ACM Symposium on Theory of Computing (STOC)*, pages 178–189, 1995. http://doi.acm.org/10.1145/225058.225112. 1, 7

[36] T. Hartman. A simpler 1.5-approximation algorithm for sorting by transpositions. In *Combinatorial pattern matching*, pages 156–169. Springer, 2003. vii, 51

[37] T. Hartman and R. Sharan. A 1.5-approximation algorithm for sorting by transpositions and transreversals. *J. Comput. Syst. Sci.*, 70(3):300–320, 2005. 8

[38] I. N. Herstein. *Abstract algebra*. Macmillan Publishing Company, 1986. 11

[39] D. Higgs and P. de Witte. On products of transpositions and their graphs. *The American Mathematical Monthly*, 86:376–380, 1979. 16, 52

[40] M. I. Honda. Implementação do algoritmo de hartman para o problema da ordenação por transposições. Master's thesis, Universidade de Brasília, 2004. vii, 51

[41] Y. L. Huang, C. C. Huang, C. Y. Tang, and C. L. Lu. An improved algorithm for sorting by block-interchanges based on permutation groups. *Information Processing Letters*, 110(8):345–350, 2010. 9

[42] J. Irving and A. Rattan. Minimal factorizations of permutations into star transpositions. *Discrete Mathematics*, 309(6):1435–1442, 2009. 17

[43] J. Kececioglu and D. Sankoff. Exact and approximation algorithms for sorting by reversals, with application to genome rearrangement. *Algorithmica*, 13(1/2):180–210, 1995. 7

[44] W. J. Kent. Blat – The BLAST-Like Alignment Tool. *Genome Research*, 12:656–664, Abril 2002. 1

[45] E. C. Lander and International Human Genome Sequencing Consortium. Initial sequencing and analysis of the human genome. *Nature*, 409(5507):860–921, February 2001. doi:10.1038/35057062. 1

[46] R. S. Lima, C. G. Ralha, H. W. Schneider, A. G. F. Pereira, M. M. Brigido, and M. E. M. T. Walter. BioAgents: Um sistema multiagente para anotação manual em projetos de seqüênciamento de genomas. In SBC, editor, *Anais do VI Encontro Nacional de Inteligência Artificial - ENIA*, pages 1302–1310, Rio de Janeiro, 2007. 1

[47] Y. C. Lin, C. L. Lu, H.-Y. Chang, and C. Y. Tang. An efficient algorithm for sorting by block-interchanges and its application to the evolution of vibrio species. *J. Comput. Biol.*, 12(1):102–112, 2005. 9

[48] G. Mackiw. Permutations as Products of Transpositions. *The American Mathematical Monthly*, 102(5):438–440, 1995. 15

[49] J. Meidanis and Z. Dias. *Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment and Evolution of Gene Families*, chapter An Alternative Algebraic Formalism for Genome Rearrangements, pages 213–223. Kluwer Academic Publishers, 2000. 8, 9, 10, 25, 52

[50] C. V. G. Mira, Z. Dias, H. P. Santos, G. A. Pinto, and M. E. M. T. Walter. Transposition distance based on the algebraic formalism. In *Advances in Bioinformatics and Computational Biology, Proceedings of the Third Brazilian Symposium on Bioinformatics*, pages 115–126, Berlin Heidelberg, Germany, 2008. Springer. 9

[51] C. V. G. Mira and J. Meidanis. Analysis of sorting by transpositions based on algebraic formalism. In *The Eighth Annual International Conference on Research in Computational Molecular Biology (RECOMB 2004)(March 2004)*, 2004. 10, 27

[52] C. V. G. Mira and J. Meidanis. Algebraic formalism for genome rearrangements (part 1). 2005. 10, 27

[53] C. V. G. Mira and J. Meidanis. Sorting by block-interchanges and signed reversals. In *Fourth International Conference on Information Technology (ITNG '07)*, pages 670–676, 2007. 9

[54] J. H. Nadeau and B. A. Taylor. Lengths of chromosomal segments conserved since divergence of man and mouse. *Proc. Natl. Acad. Sci. USA*, 81(3):814–818, 1984. 1

[55] P. C. S. Ângelo, M. M. Brígido, M. E. M. T. Walter, D. A. S. Anjos, R. M. Coimbra, and S. Astolfi-Filho. Guarana (paullinia cupana var. sorbilis), an anciently consumed stimulant from the amazon rain forest: the seeded-fruit transcriptome. *Plant Cell Reports*, 27:117–124, 2008. 1

[56] E. T. G. Oliveira. Implementações de algoritmos para o problema da distância de transposição. Master's thesis, Universidade de Brasília, 2001. vii, 51

[57] J. D. Palmer and L. A. Herbon. Plant mitochondrial dna evolves rapidly in structure, but slowly in sequence. *Journal of Molecular Evolution*, 28:87–97, 1988. 1

[58] C. G. Ralha, H. W. Schneider, L. O. Fonseca, M. E. M. T. Walter, and M. M. Brigido. Using bioAgents for supporting manual annotation on genome sequencing projects. In Springer, editor, *Advances in Bioinformatics and Computational Biology - Lecture Notes in Binformatics from the Brazilian Symposium on Bioinformatics - BSB 2008*, volume 5167, pages 127–139, Berlin - Heidelberg, 2008. 1

[59] E. Ribeiro, M. E. M. T. Walter, M. M. Costa, and G. Pappas. A Peer-to-Peer system for distributed execution of bioinformatics applications: a case study of distributed information management over massive volumes of data. In Sociedade Brasileira de Computação, editor, *Anais do CSBC 2008 - Seminário Integrado de Hardware e Software (SEMISH)*, 2008. 1

[60] E. Ribeiro, M. E. M. T. Walter, R. C. Togawa, M. M. Costa, and G. Pappas. p2pBIOFOCO: Proposing a Peer-to-Peer system for distributed BLAST execution. In IEEE, editor, *The 10th IEEE International Conference on High Performance Computing and Communications (HPCC-08)*, pages 594–601, 2008. 1

[61] J. C. Setubal and J. Meidanis. *Introduction to Computational Molecular Biology.* PWS Publishing, Estados Unidos, 1997. 1, 2

[62] M. S. Sousa and A. C. M. A. Melo. PackageBLAST: an adaptive multi-policy grid service for biological sequence comparison. In ACM, editor, *2006 ACM Symposium on Applied Computing (SAC)*, pages 156–160, Dijon, France, 2006. 1

[63] E. Tannier and M.-F. Sagot. Sorting by reversals in subquadratic time. In Springer, editor, *15th Annual Symposium on Combinatorial Pattern Matching - CPM 2004*, volume 3109, pages 1–13, Berlin - Heidelberg, 2004. 7

[64] G. P. Telles, M. M. Brigido, N. A. Franco, C. J. M. Viana, D. A. S. Anjos, and M. E. M. T. Walter. A method for comparing three genomes. In Springer, editor, *Advances in Bioinformatics and Computational Biology - Lecture Notes in Binformatics from the Brazilian Symposium on Bioinformatics - BSB 2005*, volume 3594, pages 160–1169, 2005. 1

[65] M. E. M. T. Walter. *Algoritmos em Rearranjo de Genomas*. PhD thesis, Instituto de Computação - UNICAMP, 1999. Orientador: Dr. João Meidanis. 1

[66] M. E. M. T. Walter, J. Meidanis, and Z. Dias. A new approach for aproximating the transposition distance. In *Annals of the String Processing and Information Retrieval - SPIRE'2000*, pages 279–290, 2000. vii, 8, 51

[67] G. A. Watterson, W. J. Ewens, T. E. Hall, and A. Morgan. The chromosome inversion problem. *Journal of Theoretical Biology*, 99(1):1–7, 1982. 7

[68] S. Yancopoulos, O. Attie, and R. Friedberg. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, 21(16):3340–3346, 2005. 8, 9