Este exemplar corresponde à redação final da Tese/Dissertação devidamente corrigida e defendida por: Vinicius José Fortuna

e aprovada pela Banca Examinadora.
Campinas, 28 de fevereiro de 2006

COORDENADORIA DE PÓS-GRADUAÇÃO
CPG-IC

Distâncias de Transposição entre Genomas

Vinicius José Fortuna

Dissertação de Mestrado

BIBLIOTECA CENTRAL
DESENVOLVIMENTO
COLEÇÃO
UNICAMP

Instituto de Computação
Universidade Estadual de Campinas

Distâncias de Transposição entre Genomas

Vinicius José Fortuna[1]

Fevereiro de 2005

Banca Examinadora:

- João Meidanis (Orientador)
- Maria Emília Machado Telles Walter
  CIC-UnB
- Zanoni Dias
  IC-Unicamp
- Arnaldo Vieira Moura (Suplente)
  IC-Unicamp

[1]Apoiado pela **FAPESP**, processo número **03/04578-5**.

BIBLIOTECA CENTRAL
DESENVOLVIMENTO
UNICAMP

UNIDADE BC
Nº CHAMADA
T/UNICAMP
F779d
V EX
TOMBO BC 66111
PROC. 16-P-00086-05
C D X
PREÇO 11,00
DATA 26/10/05
Nº CPD
BIBID - 366716

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP
Bibliotecário: Maria Júlia Milani Rodrigues – CRB8a / 2116

Fortuna, Vinicius José
F779d Distâncias de transposição entre genomas / Vinicius José Fortuna - - Campinas, [S.P. :s.n.], 2005.

Orientador : João Meidanis
Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação.

1. Teoria da computação. 2. Filogenia. 3. Genomas. I. Meidanis, João. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Título em inglês: Transposition distances between genomes

Palavras-chave em inglês (Keywords): 1. Computer theory. 2. Phylogeny. 3. Genomes

Área de concentração: Biologia computacional

Titulação: Mestre em Ciência da Computação

Banca examinadora: Prof. Dr. João Meidanis (IC-UNICAMP)
Profa. Dra. Maria Emília Machado Telles Walter (CIC-UnB)
Prof. Dr. Zanoni dias (IC-UNICAMP)

Data da defesa: 28/03/2005.

Distâncias de Transposição entre Genomas

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Vinicius José Fortuna e aprovada pela Banca Examinadora.

Campinas, 28 de março de 2005.

João Meidanis (Orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

TERMO DE APROVAÇÃO

Tese defendida e aprovada em 28 de março de 2005, pela Banca examinadora composta pelos Professores Doutores:

Profa. Dra. Maria Emilia Machado Telles Walter
UnB

Prof. Dr. Zanoni Dias
IC - UNICAMP

Prof. Dr. João Meidanis
IC - UNICAMP

Resumo

Uma das principais formas de se medir a distância evolutiva entre espécies é avaliando-se quão transformado um genoma foi em relação a outro. Tais transformações são conhecidas como *rearranjos de genoma*. Neste trabalho estaremos analisando o rearranjo chamado de *transposição*, evento que troca de posição dois blocos consecutivos de genes de um mesmo cromossomo. Mais especificamente, buscamos encontrar o número mínimo de transposições que transforma um cromossomo em outro, valor conhecido como distância de transposição. Matematicamente, consideramos os cromossomos como permutações e o problema de se transformar uma permutação em outra pode ser visto como uma ordenação.

Em nosso estudo, introduzimos uma operação de remoção de elementos, ferramenta ainda pouco explorada no estudo da distância de transposição, mas que nos possibilitou obter um limite superior para a distância de transposição. Também sugerimos novas formas de se utilizar a remoção de elementos e a análise de subseqüências de permutações.

Como forma de se tentar obter novos conhecimentos sobre o problema da distância de transposição, consideramos a variação do problema em que somente *transposições de prefixo* são permitidas. Zanoni Dias propôs um algoritmo polinomial que transforma qualquer permutação de comprimento n em sua reversa em $\lceil 3n/4 \rceil$ passos, porém sem uma prova completa. Modificamos esse algoritmo, mantendo o número de passos, e apresentamos uma prova completa da correção do algoritmo modificado.

Ainda no problema de distância de transposição de prefixo, analisamos as permutações cujas distâncias se igualam ao limite inferior de distância de pontos de quebra. Tais permutações são fáceis de serem ordenadas por transposições de prefixo em tempo polinomial, pois possuem ordenação ótima única e bem definida. Ao final chegamos à conclusão que as permutações que são fáceis de se ordenar no problema de transposições de prefixo também são fáceis no problema de transposições, o que prova que a variação do problema auxilia no estudo do problema original.

Abstract

One of the main ways of measuring the evolution distance among species is to evaluate how large chunks have moved when comparing two genomes. Such changes are know as *genome rearrangements*. In this work we analyze a rearrangement event called *transposition* that changes the position of two consecutive blocks of genes in a chromosome. More specifically, we look for the minimum number of transpositions needed to transform a chromosome into another. This value is called the transposition distance. Mathematically, chromosomes are regarded as permutations and changing one genome into another can be seen as a sorting problem.

In our study, we introduce an operation of element removal from permutations, which has not been fully explored, but allowed us to find an upper bound for the transposition distance. We also suggest new ways of making use of element removal and the analysis of permutation subsequences.

In the hope of obtaining new knowledge about the problem of transposition distance, we considered the variation of the problem where we allow prefix transpositions only. Zanoni Dias developed a polynomial algorithm that changes any permutation into its reverse using $\lceil 3n/4 \rceil$ steps, but without a proof of its correctness. We have modified this algorithm, keeping the number of steps, and presented a complete proof of the correction of the modified algorithm.

Still about the prefix transposition distance, we have analyzed those permutations whose distance equals the breakpoint lower bound. Such permutations are easily sorted by prefix transpositions in polynomial time, since they have a unique and well-defined optimum sorting. Finally, we concluded that the permutations that are easily sorted in the prefix transposition problem are also easily sorted in the transposition problem, which proves that the variation of the problem helps the study of the original problem.

Agradecimentos

Inicialmente gostaria de agradecer aos meus queridos pais Vilmar e Ana Dalva, que sempre me deram total apoio para chegar até aqui. Se não fosse pelo incentivo, ajuda e carinho que eles tiveram comigo em todo esse tempo provavelmente não estaria tão bem. Pai, mãe, esse trabalho é pra vocês!

Agradeço também ao meu orientador, João Meidanis, que me ensinou muita coisa nesses anos, me passando diversos ensinamentos, não apenas sobre a área acadêmica, mas também sobre a vida.

À minha querida Giselli devo um agradecimento especial pela paciência e compreensão que teve nos momentos que tive que trocá-la pelo computador ou pelas fórmulas matemáticas para o desenvolvimento deste trabalho e pelo amor e carinho com que me acolheu nos momentos que precisei dela.

Também devo agradecer à minha tia Marlene e minha avó Olimpia, que foram sempre super preocupadas comigo e me ajudaram a levar a vida em Campinas, além de sempre torcerem muito pelo meu sucesso, assim como meus irmãos Vitor e Frederico. Vale também a lembrança de meu tio Luiz Paulo, que certamente está muito feliz nesse momento.

Não posso esquecer de agradecer aos meus amigos, que tornaram minha vida em Campinas tão agradável, especialmente os mais próximos: Ulisses, Tomita, Sheila, que também compartilharam comigo do mesmo "drama" e dificuldades de cursar as disciplinas e escrever relatórios, artigos e a dissertação de mestrado, onde um sempre procurava incentivar e orientar o outro.

Sou muito grato aos professores e funcionários do Instituto de Computação da Unicamp, que nos ensinaram e ajudaram nas questões do dia-a-dia, em especial ao professor Setubal, que me orientou diante das dúvidas no início do meu mestrado.

Sou grato também à FAPESP, que apoiou financeiramente a minha pesquisa, o que permitiu que eu me dedicasse com mais afinco às tarefas acadêmicas.

Enfim, agradeço a todos aqueles que contribuíram, mesmo que um pouquinho, com uma palavra, um pensamento, para o desenvolvimento do meu trabalho e a todos aqueles que torcem pelo meu sucesso e estão felizes com essa minha realização.

Sumário

Resumo vii

Abstract viii

Agradecimentos ix

1 Introdução 1
1.1 Contexto . . . 1
1.2 Definições e Resultados Básicos . . . 4
1.2.1 Genomas e Permutações . . . 4
1.2.2 Operações de Transposição . . . 5
1.2.3 Distância e Diâmetro . . . 6
1.2.4 Pontos de Quebra . . . 6
1.2.5 Ciclos . . . 11

2 Distância de Transposição 14
2.1 Introdução . . . 14
2.2 Limite Inferior . . . 15
2.3 Limite Superior . . . 16
2.4 Conclusões e Problemas em Aberto . . . 19

3 Distância de Transposição de Prefixo 22
3.1 Introdução . . . 22
3.2 Ordenação da Permutação Reversa . . . 22
3.2.1 Algoritmo para Ordenar a Permutação Reversa . . . 23
3.3 Permutações Fáceis . . . 41
3.3.1 Propriedades . . . 42
3.3.2 Contagem . . . 46
3.4 Conclusões e Problemas em Aberto . . . 48

4 Conclusão 51

A Implementações 53
A.1 *Mathematica* 53
A.2 Distribuição Distância × Pontos de quebra 56

Bibliografia 63

Lista de Figuras

1.1 Diagrama de ciclos para $\pi = [4,\ 2,\ 6,\ 1,\ 5,\ 3]$. 12

3.1 Ordenação da permutação reversa. 23
3.2 Razão entre $\mathcal{F}(n)$ e $n!$. 49

A.1 Uso da função `PrintP`. 55
A.2 Diagrama de ciclos das permutações intermediárias na ordenação de $F_1(8)$. 56
A.3 Grafo de distâncias de transposição de prefixo. 57

Capítulo 1

Introdução

1.1 Contexto

Esta dissertação apresenta um trabalho teórico em Biologia Computacional, área que utiliza técnicas computacionais para resolver problemas biológicos. A Biologia Computacional cresceu muito rapidamente na ultima década pela necessidade de processar a enorme massa de dados genéticos que passou a ser produzida com o advento de técnicas de sequenciamento rápido de genomas.

Podemos citar como exemplo de problema biológico resolvido pela Biologia Computacional a busca por genes comuns a duas espécies. Uma forma de se resolver esse problema em Biologia seria por meio de diversos experimentos em laboratório. Com a ajuda da Biologia Computacional podemos resolver de uma forma muito mais eficiente. Uma vez encontradas as seqüências de nucleotídeos que compõem dois trechos de moléculas de DNA, podemos descrevê-las como cadeias de caracteres do alfabeto {A,C,G,T}. O problema passa então a ser de comparação de strings. Onde encontramos trechos semelhantes há uma maior probabilidade deles descreverem um gene comum. Cabe ressaltar que um mesmo gene em uma espécie pode aparecer ligeiramente diferente em outra espécie devido a pequenas mutações locais, como inserção, remoção e substituição. Por isso a busca pelos trechos em comum deve ser feita tolerando pequenas diferenças. Veja o livro texto de Pevzner [25] ou o de Setubal e Meidanis [26] para maiores detalhes sobre como fazer isso.

Um outro problema biológico de grande relevância é a classificação filogenética das espécies e a descrição de suas cadeias evolutivas. Duas espécies muito próximas possuem grande parte de seus códigos genéticos em comum. Os genes mitocondriais humanos, por exemplo, são mais de 91% idênticos aos dos chimpanzés [27]. Genes que aparecem em espécies diferentes, mas que têm a mesma origem são chamados de *ortólogos*. Espécies com muitos genes ortólogos entre si possuem um ancestral comum próximo e são, portanto,

evolutivamente próximas.

Podemos ainda inferir as relações evolutivas entre espécies por meio da ordem em que os genes ortólogos aparecem em seus cromossomos [24]. Isso porque esses genes vieram de um ancestral comum e, portanto, mudaram de posição durante o processo evolutivo até as espécies em questão. Essas mudanças na posição dos genes são conhecidas como *rearranjos de genoma*. Assumindo que a natureza é parcimoniosa, devemos utilizar a explicação mais simples para a transformação de um genoma em outro, que é a transformação com o menor número de rearranjos. Esse valor é chamado *distância de rearranjo*.

Os eventos de rearranjo mais significativos na natureza são reversões e transposições [2, 4]. Uma reversão inverte a ordem e a orientação de um bloco de genes em um cromossomo do genoma. Uma transposição remove um bloco de genes de um cromossomo e insere-o em uma nova posição no mesmo cromossomo. Existem ainda rearranjos que removem um trecho de um cromossomo e insere-o em outro, revertido ou não.

O cálculo da distância de rearranjo considerando apenas reversões e transposições foi estudado por alguns cientistas [28, 15, 20, 11], mas mostrou-se muito difícil, sendo um problema em aberto até hoje. Como forma de se contornar essa dificuldade os rearranjos são estudados separadamente, surgindo, por exemplo, os problemas de cálculo da distância de reversão e da distância de transposição, que são, respectivamente, o menor número de reversões ou transposições necessárias para transformar um genoma em outro.

No caso de reversões e transposições, transformar um genoma em outro é colocar os genes dos cromossomos do primeiro na mesma ordem dos do segundo. Por isso os problemas de se calcular a distância de reversão ou de transposição podem ser vistos como problemas de ordenação por meio de operações de reversão ou de transposição, respectivamente. É importante notar que sempre é possível transformar um cromossomo em outro com o mesmo conjunto de genes por meio de reversões. Isso também vale para transposições caso os genes possuam a mesma orientação em ambos os cromossomos.

O problema de distância de reversão já foi muito bem explorado. Se desconsiderarmos a orientação dos genes, o problema foi provado NP-difícil [7] e surgiram diversos algoritmos aproximados [3, 6, 8]. Por outro lado, considerando a orientação dos genes, o problema pode ser resolvido em tempo polinomial, como evidenciado pelo algoritmo de Hannenhalli e Pevzner [17]. Trabalhos mais recentes [1, 5, 19] simplificaram a teoria e melhoraram o tempo de execução do algoritmo. Também foi provado que o problema da distância de reversão para cromossomos circulares é equivalente ao problema com cromossomos lineares [23].

Com o problema do cálculo da distância de transposição houve menos progresso. Até hoje não se sabe se o problema é NP-difícil ou não. Ele foi inicialmente estudado por Bafna e Pevzner [4], que desenvolveram um algoritmo cúbico com fator de aproximação 1.5. Em seguida Christie [9] desenvolveu um algoritmo quadrático com mesmo fator

de aproximação. Ambos os algoritmos eram muito complicados, baseados em noções complexas sobre o *grafo de ciclos*. Walter, Dias e Meidanis [29] sugeriram um algoritmo de 2.25-aproximação, só que mais simples, baseado no conceito mais básico de *diagrama de pontos de quebra*. Eriksson e colegas [12] desenvolveram um algoritmo que precisa de no máximo $\lfloor \frac{2n-1}{3} \rfloor$ transposições para transformar um cromossomo com n genes em outro qualquer com os mesmos genes na mesma orientação. Entretanto esse algoritmo não possui garantia de aproximação. Mais recentemente, Hartman e Shamir [18] apresentaram um novo algoritmo com fator de aproximação 1.5 mais simples e mais rápido, com uma complexidade de tempo $O(n^{3/2}\sqrt{\log n})$.

Apesar desses resultados, o problema da distância de transposição continua em aberto. Como forma de se tentar obter resultados e visões diferentes sobre o problema, Dias e Meidanis [10] propuseram uma nova variação do problema, onde somente são permitidas transposições que afetam o primeiro elemento da permutação.

Nosso objetivo com essa dissertação é prosseguir com o estudo do problema da distância de transposição, tentando novas abordagens e variações do problema.

Apresentaremos o trabalho realizado da seguinte forma. Como continuação da introdução apresentaremos as definições dos conceitos mais básicos utilizados no estudo da distância de transposição e alguns resultados relevantes obtidos anteriormente e que nos ajudarão ao longo do trabalho. Os capítulos seguintes são todos de contribuições originais.

O Capítulo 2 apresenta operações e idéias novas relacionadas ao conceito de subseqüência, assunto abordado anteriormente por Guyer, Heath e Vergara [16], mas ainda pouco explorado para o estudo da distância de transposição, e obtemos alguns resultados que mostram seu potencial. Concluímos sugerindo outras possibilidades de se explorar as idéias apresentadas.

No Capítulo 3 estudamos a variação do problema que considera apenas transposições de prefixo como forma de obter outros pontos de vista sobre o problema original. Esse capítulo é dividido em duas partes principais. Na primeira apresentaremos uma forma de se transformar um cromossomo em outro com seus genes na ordem inversa, pois acredita-se que essa seja a transformação que exige mais transposições e, com isso, tentamos encontrar quais as propriedades tornam uma transformação por transposições de prefixo difícil. Na segunda parte vemos a situação inversa. Estudamos quais cromossomos podem ser transformados em outro com o número mínimo de transposições e, com isso, tentamos encontrar quais as propriedades tornam uma transformação por transposições de prefixo fácil.

Por fim apresentamos uma conclusão que resume os resultados obtidos e propõe trabalhos futuros, seguida de um apêndice que ilustra uma parte das ferramentas computacionais implementadas para auxiliar no trabalho desta dissertação de mestrado.

1.2 Definições e Resultados Básicos

Vista a motivação biológica do estudo de rearranjo, em especial as transposições, precisamos modelar o problema matematicamente para podermos aplicar argumentos formais e assim tentar resolvê-lo.

1.2.1 Genomas e Permutações

Trataremos genomas como sendo formados por cromossomos únicos lineares e sem genes repetidos, sendo então representados por permutações. Uma permutação é qualquer ordenação dos elementos de um conjunto finito. Uma permutação dos elementos de um conjunto de tamanho n é chamada de *permutação de comprimento* n. O i-ésimo elemento de uma permutação π será representado por $\pi(i)$. Permutações serão representadas por seus elementos entre colchetes e separados por vírgulas.

Exemplo Se π é qualquer permutação de comprimento n, então π=$[\pi(1), \pi(2), \ldots, \pi(n-1), \pi(n)]$.

Permutações podem ser vistas como funções, de maneira que uma permutação π dos elementos de um conjunto E de tamanho n seja uma bijeção $\pi : [1, n] \rightarrow E$. Dessa forma faz sentido falar na inversa π^{-1} da permutação π, que é uma bijeção que, dado um elemento do conjunto E, retorna sua posição na permutação π.

Para facilitar a generalização de alguns conceitos, estenderemos essa definição de forma que $\pi(0) = 0$ para toda permutação π. É importante que o elemento 0 (zero) não pertença ao conjunto de genes da permutação π. Caso pertença, basta renomear o gene.

Uma outra vantagem de ver permutações como funções é que podemos fazer composições entre elas, obtendo como resultado, uma *renomeação* ou *reordenação* dos elementos. Se temos uma permutação $\pi : [1, n] \rightarrow E$, aplicar uma permutação $\phi : [1, n] \rightarrow [1, n]$ pela direita, obtendo $\pi\phi$, significa reordenar os elementos de π segundo ϕ. Neste caso, $\pi\phi(i) = \pi(\phi(i))$, ou seja, o i-ésimo elemento da permutação $\pi\phi$ é o $\phi(i)$-ésimo elemento da permutação π.

Exemplo Se $\pi = [a, b, c, d, e, f]$ e $\phi = [3, 2, 4, 1, 6, 5]$, então

$$\pi\phi = [\pi(3), \pi(2), \pi(4), \pi(1), \pi(6), \pi(5)] = [c, b, d, a, f, e].$$

Perceba que $\pi\phi$ equivale a pegar os elementos de π na ordem de ϕ.

Se temos uma permutação $\pi : [1, n] \rightarrow E$, aplicar uma função $\rho : E \rightarrow S$ pela esquerda, obtendo $\rho\pi$, significa renomear os elementos de π segundo ρ. Neste caso, $\rho\pi(i) = \rho(\pi(i))$, ou seja, o i-ésimo elemento da permutação $\rho\pi$ é o elemento $\pi(i)$ renomeado de acordo com ρ.

Exemplo Se $\pi = [a, b, c, d, e, f]$ e $\rho(a) = p$, $\rho(b) = q$, $\rho(c) = r$, $\rho(d) = s$, $\rho(e) = t$ e $\rho(f) = u$, então $\rho\pi = [p, q, r, s, t, u]$. Da mesma forma, $\pi^{-1}\pi = [1, 2, 3, 4, 5, 6]$.

1.2.2 Operações de Transposição

Uma *transposição* é uma operação sobre uma permutação que remove um bloco de elementos e reinsere-o em uma nova posição. Tal operação pode ser vista como a troca de posição entre dois blocos adjacentes. Note que desfazer uma transposição também é uma transposição. Por se tratar de uma reordenação, o resultado da aplicação de uma transposição τ a uma permutação π será representado por $\pi\tau$.

Exemplo Seja $\pi = [a, b, c, d, e, f, g]$ e τ uma transposição que remove os elementos nas posições 2 e 3 e insere-os de volta antes do último elemento. Então $\pi\tau = [a, d, e, f, b, c, g]$. Observe que tal transposição também pode ser vista como a troca de posição entre o bloco de elementos $[b, c]$ com o bloco de elementos $[d, e, f]$.

No exemplo anterior utilizamos linguagem textual para definir qual seria a transposição a ser aplicada. Entretanto isso é muitas vezes inconveniente em fórmulas e em provas de teoremas. Precisamos, portanto, de uma notação matemática para definir uma transposição. Utilizaremos a expressão $\tau(i, j, k)$, onde $1 \leqslant i < j < k \leqslant n + 1$, para se referir à transposição que troca de posição o bloco dos elementos nas posições i a $j - 1$ com o bloco de elementos nas posições j a $k - 1$.

Utilizando a definição de reordenação apresentada anteriormente, $\tau(i, j, k)$, quando aplicada à uma permutação de comprimento n, é uma permutação $\tau : [1, n] \rightarrow [1, n]$ tal que $\tau = [1, \ldots, i - 1, j, \ldots, k - 1, i, \ldots, j - 1, k, \ldots, n]$.

Exemplo Considere a permutação

$$\pi = [\pi(1), \ldots, \pi(i - 1), \pi(i), \ldots, \pi(j - 1), \pi(j), \ldots, \pi(k - 1), \pi(k), \ldots, \pi(n)].$$

Utilizando a notação apresentada temos que

$$\pi\tau(i, j, k) = [\pi(1), \ldots, \pi(i - 1), \pi(j), \ldots, \pi(k - 1), \pi(i), \ldots, \pi(j - 1), \pi(k), \ldots, \pi(n)].$$

Um caso específico das transposições é quando um dos blocos que trocam de posição inclui o primeiro elemento da permutação. Chamaremos tal transposição de *transposição de prefixo*, pois estamos removendo um prefixo da permutação e inserindo-o em uma nova posição. Em notação matemática, as transposições de prefixo são todas aquelas na forma $\tau(1, i, j)$.

Existem outras formas de se definir e representar permutações e operações sobre elas no contexto de rearranjo de genomas. Um bom exemplo é a Teoria Algébrica de Dias e Meidanis [21], que permite dar um tratamento algébrico aos problemas de rearranjo. Contudo, tal abordagem não será utilizada neste trabalho.

1.2.3 Distância e Diâmetro

Transformar por transposições uma permutação em outra é aplicar transposições à primeira de forma que a permutação resultante seja igual à segunda. A *distância de transposição* $d(\pi, \sigma)$ entre duas permutações π e σ é o número mínimo de transposições necessário para transformar π em σ. É importante notar que essa distância será sempre finita se ambas as permutações forem de um mesmo conjunto de elementos. Caso contrário a distância será infinita, pois não é possível transformar uma permutação em outra com elementos diferentes apenas com transposições. O *diâmetro de transposição* $D(n)$ é a maior distância de transposição possível entre duas permutações de comprimento n sobre um mesmo conjunto de elementos.

De forma análoga, a *distância de transposição de prefixo* $d_p(\pi, \sigma)$ entre duas permutações π e σ é o número mínimo de transposições de prefixo necessário para transformar π em σ. O *diâmetro de transposição de prefixo* $D_p(n)$ é a maior distância de transposição de prefixo possível entre duas permutações de comprimento n. Obviamente, para todo par de permutações π, σ, temos $d(\pi, \sigma) \leqslant d_p(\pi, \sigma)$, já que toda transposição de prefixo é uma transposição. Com isso também garantimos $D(n) \leqslant D_p(n)$.

Se temos uma permutação π dos elementos de um conjunto E, e esses elementos possuem uma ordem natural, então *ordenar π por transposições* consiste em aplicar transposições a π de forma que a permutação resultante possua todos os seus elementos em sua ordem natural. A distância de transposição $d(\pi)$ de uma única permutação π é o número mínimo de transposições necessário para ordená-la. Não é difícil perceber que a distância $d(\pi)$ é equivalente a $d(\pi, \sigma)$, onde σ é uma permutação com todos seus elementos em sua ordem natural.

Se consideramos apenas transposições de prefixo, também podemos definir, de forma análoga, *ordenação por transposições de prefixo* e distância de transposição de prefixo $d_p(\pi)$ de uma única permutação π.

1.2.4 Pontos de Quebra

No estudo de rearranjos de genomas, queremos medir quão modificada uma permutação ficou em relação à outra devido aos eventos de rearranjos. Uma medida aproximada de quão desordenados estão os elementos de uma permutação com relação a outra é o número de *pontos de quebra*, que indica o número de pares de elementos que são adjacentes na permutação destino, mas que não são adjacentes na permutação origem.

Definiremos o *predecessor* $pred(e, \pi)$ de um elemento $e \in E$ em uma permutação π dos elementos de E como sendo o elemento que vem antes de e na permutação π, ou seja $\pi(\pi^{-1}(e)-1)$. Definiremos o *sucessor* $suc(e, \pi)$ de um elemento $e \in E$ em uma permutação π dos elementos de E como sendo o elemento que vem após e na permutação π, ou seja

$\pi(\pi^{-1}(e)+1)$. Com isso, $suc(pred(e,\pi),\pi) = e$, para todo $e \in E$. Definiremos também $pred^k(e,\pi) = \pi(\pi^{-1}(e)-k)$ como sendo o k-ésimo predecessor e $suc^k(e,\pi) = \pi(\pi^{-1}(e)+k)$ como sendo o k-ésimo sucessor. Para garantir que estas definições façam sentido para qualquer valor inteiro de k, vamos sempre supor que as operações de adição e subtração são feitas módulo $n+1$, onde $n = |E|$, e considerar a definição estendida de permutação, de forma que $\pi(0) = 0$.

Também definimos essas funções com apenas um parâmetro, de forma semelhante ao que fizemos com a distância. Se σ é a permutação dos elementos de π em sua ordem natural, então $pred(\pi) = pred(\pi,\sigma)$, $pred^k(\pi) = pred^k(\pi,\sigma)$, $suc(\pi) = suc(\pi,\sigma)$ e $suc^k(\pi) = suc^k(\pi,\sigma)$.

Os pontos de quebra de uma permutação π em relação à uma permutação σ são todos aqueles elementos e tais que $pred(e,\pi) \neq pred(e,\sigma)$. Assim, o conjunto de pontos de quebra de uma permutação π em relação à uma permutação σ, ambas sobre um conjunto E, é dado por

$$B(\pi,\sigma) = \{e \in E \cup \{0\} \mid pred(e,\pi) \neq pred(e,\sigma)\}.$$

O número de pontos de quebra de uma permutação π em relação à uma permutação σ é dado por $b(\pi,\sigma) = |B(\pi,\sigma)|$. Observe que $b(\sigma,\sigma) = 0$.

Também definimos essas funções com apenas um parâmetro, de forma semelhante ao que fizemos com a distância. Se σ é a permutação dos elementos de π em sua ordem natural, então $B(\pi) = B(\pi,\sigma)$ e $b(\pi) = b(\pi,\sigma)$.

É importante notar o seguinte lema.

Lema 1.1 *Para todo par de permutações π e σ, se $\pi \neq \sigma$ então $b(\pi,\sigma) \geqslant 3$.*

Prova: Se $\pi \neq \sigma$, então existe um $e \in E$ tal que $pred(e,\pi) \neq pred(e,\sigma)$. Mas se isso é verdade então podemos exibir dois outros elementos $f, g \in E$ tais que $pred(f,\pi) \neq pred(f,\sigma)$ e $pred(g,\pi) \neq pred(g,\sigma)$. Para isso basta pegar $f = suc(pred(e,\pi),\sigma)$, pois $pred(f,\sigma) = pred(e,\pi) \neq pred(f,\pi)$ e $g = suc(pred(e,\sigma),\pi)$, pois $pred(g,\pi) = pred(e,\sigma) \neq pred(g,\sigma)$.

Note também que $pred(f,\sigma) \neq pred(e,\sigma)$ e $pred(g,\pi) \neq pred(e,\pi)$, portanto $f \neq e$ e $g \neq e$. Se $g \neq f$ ou houver outro ponto de quebra além de e, f e g, então o problema está resolvido. Caso contrário, devemos analisar a possibilidade $f = g$.

Sabemos que existe um k entre 1 e n tal que

$$e = pred^k(f,\pi) = pred^{k-1}(pred(f,\pi),\pi) = pred^{k-1}(pred(e,\sigma),\pi).$$

Como não há outros pontos de quebra, $e = pred^{k-1}(pred(e,\sigma),\sigma) = pred^k(e,\sigma)$, o que é absurdo. Portanto $g \neq f$ e temos pelo menos três pontos de quebra. ■

O valor $b(\pi, \sigma)$ também pode ser visto como um somatório. Para isso, inspirados no livro de Graham e colegas [14], definiremos que $[[\varepsilon]]$ vale 1, se a expressão booleana ε for verdadeira, e 0 se falsa. Temos então

$$b(\pi, \sigma) = \sum_{e \in E \cup \{0\}} [[pred(e, \pi) \neq pred(e, \sigma)]].$$

Com isso podemos facilmente calcular a variação no número de pontos de quebra $b(\pi, \sigma)$ devida à aplicação de uma transposição $\tau(i, j, k)$ a π. Note que com essa transposição, apenas os predecessores de $\pi(i)$, $\pi(j)$ e $\pi(k)$ se alteram. Dessa forma obtemos

$$\begin{aligned}
&b(\pi\tau, \sigma) - b(\pi, \sigma) \\
&\quad = \sum_{e \in E \cup \{0\}} [[pred(e, \pi\tau) \neq pred(e, \sigma)]] - \sum_{e \in E \cup \{0\}} [[pred(e, \pi) \neq pred(e, \sigma)]] \\
&\quad = \sum_{e \in E \cup \{0\}} ([[pred(e, \pi\tau) \neq pred(e, \sigma)]] - [[pred(e, \pi) \neq pred(e, \sigma)]]) \\
&\quad = [[pred(\pi(i), \pi\tau) \neq pred(\pi(i), \sigma)]] - [[pred(\pi(i), \pi) \neq pred(\pi(i), \sigma)]] + \\
&\qquad [[pred(\pi(j), \pi\tau) \neq pred(\pi(j), \sigma)]] - [[pred(\pi(j), \pi) \neq pred(\pi(j), \sigma)]] + \\
&\qquad [[pred(\pi(k), \pi\tau) \neq pred(\pi(k), \sigma)]] - [[pred(\pi(k), \pi) \neq pred(\pi(k), \sigma)]].
\end{aligned}$$

Simplificando os termos e negando as expressões booleanas, obtemos a seguinte proposição.

Proposição 1.2 *Para toda transposição τ e par de permutações π e σ, temos*

$$\begin{aligned}
b(\pi\tau, \sigma) - b(\pi, \sigma) \quad = \quad & [[\pi(i-1) = pred(\pi(i), \sigma)]] - [[\pi(k-1) = pred(\pi(i), \sigma)]] + \\
& [[\pi(j-1) = pred(\pi(j), \sigma)]] - [[\pi(i-1) = pred(\pi(j), \sigma)]] + \\
& [[\pi(k-1) = pred(\pi(k), \sigma)]] - [[\pi(j-1) = pred(\pi(k), \sigma)]].
\end{aligned}$$

Com isso fica claro o seguinte lema.

Lema 1.3 *Para toda transposição τ e permutações π e σ, temos*

$$-3 \leqslant b(\pi\tau, \sigma) - b(\pi, \sigma) \leqslant 3.$$

O que nos permite provar um limite inferior para a distância de transposição.

Teorema 1.4 (Bafna e Pevzner [4]) *Para todo par de permutações π e σ, temos*

$$d(\pi, \sigma) \geqslant \frac{b(\pi, \sigma)}{3}.$$

Prova: Para transformar π em σ precisamos tirar todos os $b(\pi,\sigma)$ pontos de quebra. Pelo Lema 1.3 não podemos retirar mais do que 3 pontos de quebra por transposição, então precisamos de pelo menos $\left\lceil \frac{b(\pi,\sigma)}{3} \right\rceil$ transposições. ■

No contexto do problema da distância de transposição de prefixo entre duas permutações π e σ, não há vantagem nenhuma em ter o primeiro elemento de σ como o primeiro elemento de π, já que temos sempre que retirar o primeiro elemento de seu lugar ao aplicar uma transposição de prefixo. Dessa forma passa a ser interessante considerar $\pi(1)$ como um ponto de quebra mesmo que $pred(\pi(1),\pi) = pred(\pi(1),\sigma)$.

Com isso definiremos que os *pontos de quebra de prefixo* de uma permutação π em relação à uma permutação σ são todos aqueles elementos e tais que $pred(e,\pi) \neq pred(e,\sigma)$ ou $e = \pi(1)$.

O conjunto de pontos de quebra de prefixo de uma permutação π em relação à uma permutação σ, ambas sobre um conjunto E, é dado por

$$B_p(\pi,\sigma) = \{e \in E \mid pred(e,\pi) \neq pred(e,\sigma) \text{ ou } e = \pi(1)\}.$$

O número de pontos de quebra de prefixo de uma permutação π em relação à uma permutação σ é dado por $b_p(\pi,\sigma) = |B_p(\pi,\sigma)|$. Observe que $b_p(\sigma,\sigma) = 1$.

Também definimos essas funções com apenas um parâmetro. Se σ é a permutação dos elementos de π em sua ordem natural, então $B_p(\pi) = B_p(\pi,\sigma)$ e $b_p(\pi) = b_p(\pi,\sigma)$.

Podemos relacionar o número de pontos de quebra $b(\pi,\sigma)$ com o número de pontos de quebra de prefixo $b_p(\pi,\sigma)$ da seguinte forma:

$$\begin{aligned}
b_p(\pi,\sigma) &= |\{e \in E \mid pred(e,\pi) \neq pred(e,\sigma)\} \cup \{\pi(1)\}| \\
&= |\{e \in E \mid pred(e,\pi) \neq pred(e,\sigma)\}| + |\{\pi(1)\}| - \\
&\quad |\{e \in E \mid pred(e,\pi) \neq pred(e,\sigma) \text{ e } e = \pi(1)\}| \\
&= b(\pi,\sigma) + 1 - [[0 \neq pred(\pi(1),\sigma)]] \\
&= b(\pi,\sigma) + 1 - [[\sigma(1) \neq \pi(1)]] \\
&= b(\pi,\sigma) + [[\pi(1) = \sigma(1)]].
\end{aligned}$$

De forma análoga ao Lema 1.1, temos o seguinte lema, que garante que não existe um par de permutações π e σ com exatamente 2 pontos de quebra ou com exatamente 3 pontos de quebra e com $\pi(1) = \sigma(1)$.

Lema 1.5 *Para todo par de permutações π e σ, se $\pi \neq \sigma$ então $b_p(\pi,\sigma) \geqslant 3 + [[\pi(1) = \sigma(1)]]$.*

Prova: Pelo Lema 1.1, temos $b(\pi,\sigma) \geqslant 3$. Substituindo $b(\pi,\sigma)$ por $b_p(\pi,\sigma) - [[\pi(1) = \sigma(1)]]$, obtemos $b_p(\pi,\sigma) \geqslant 3 + [[\pi(1) = \sigma(1)]]$. ■

Ainda utilizando a relação entre b e b_p temos, para toda transposição de prefixo $\tau(1,j,k)$ e permutações π e σ, a seguinte variação no número de pontos de quebra de prefixo

$$\begin{aligned} b_p(\pi\tau,\sigma) - b_p(\pi,\sigma) &= b(\pi\tau,\sigma) + [[\pi\tau(1) = \sigma(1)]] - b(\pi,\sigma) - [[\pi(1) = \sigma(1)]] \\ &= [[\pi(0) = pred(\pi(1),\sigma)]] - [[\pi(k-1) = pred(\pi(1),\sigma)]] + \\ &\quad [[\pi(j-1) = pred(\pi(j),\sigma)]] - [[\pi(0) = pred(\pi(j),\sigma)]] + \\ &\quad [[\pi(k-1) = pred(\pi(k),\sigma)]] - [[\pi(j-1) = pred(\pi(k),\sigma)]] + \\ &\quad [[\pi\tau(1) = \sigma(1)]] - [[\pi(1) = \sigma(1)]] \\ &= [[\pi(1) = \sigma(1)]] - [[\pi(k-1) = pred(\pi(1),\sigma)]] + \\ &\quad [[\pi(j-1) = pred(\pi(j),\sigma)]] - [[\sigma(1) = \pi(j)]] + \\ &\quad [[\pi(k-1) = pred(\pi(k),\sigma)]] - [[\pi(j-1) = pred(\pi(k),\sigma)]] + \\ &\quad [[\pi\tau(1) = \sigma(1)]] - [[\pi(1) = \sigma(1)]]. \end{aligned}$$

Simplificando a expressão, obtemos a seguinte proposição.

Proposição 1.6 *Para toda transposição de prefixo $\tau(1,j,k)$ e par de permutações π e σ, temos*

$$\begin{aligned} b_p(\pi\tau,\sigma) - b_p(\pi,\sigma) &= [[\pi(j-1) = pred(\pi(j),\sigma)]] - [[\pi(j-1) = pred(\pi(k),\sigma)]] + \\ &\quad [[\pi(k-1) = pred(\pi(k),\sigma)]] - [[\pi(k-1) = pred(\pi(1),\sigma)]]. \end{aligned}$$

Com isso fica claro o seguinte lema.

Lema 1.7 *Para toda transposição de prefixo τ e permutações π e σ, temos*

$$-2 \leqslant b_p(\pi\tau,\sigma) - b_p(\pi,\sigma) \leqslant 2.$$

O que nos permite provar um limite inferior para a distância de transposição de prefixo.

Teorema 1.8 (Dias e Meidanis [10]) *Para todo par de permutações π e σ, temos*

$$d_p(\pi,\sigma) \geqslant \frac{b_p(\pi,\sigma) - 1}{2}.$$

Prova: Para transformar π em σ precisamos retirar os $b_p(\pi,\sigma)$ pontos de quebra de prefixo e deixar apenas um. Pelo Lema 1.7 não podemos retirar mais do que 2 pontos de quebra de prefixo por transposição de prefixo, então precisamos de pelo menos $\left\lceil \frac{b_p(\pi,\sigma)-1}{2} \right\rceil$ transposições de prefixo. ■

Para obter um limite superior, precisamos provar mais alguns lemas.

Lema 1.9 *Dadas duas permutações π e σ tais que $\pi \neq \sigma$, sempre é possível obter uma transposição de prefixo τ tal que $b_p(\tau\pi, \sigma) - b_p(\pi, \sigma) \leqslant -1$.*

Prova: Seja $e = \pi(i)$ o primeiro ponto de quebra tal que $i > 1$. Este elemento existe pois $\pi \neq \sigma$. Temos que $f = suc(pred(e, \pi), \sigma))$ também é um ponto de quebra. Basta aplicar a transposição $\tau(1, \pi^{-1}(e), \pi^{-1}(f))$ para f deixar de ser ponto de quebra sem criar nenhum ponto de quebra novo, o que pode ser confirmado usando-se a Proposição 1.6. ■

Lema 1.10 *Seja π e σ permutações tais que $b_p(\pi, \sigma) = 3$. Então $d_p(\pi, \sigma) = 1$.*

Prova: Pelo Lema 1.9, existe uma transposição de prefixo τ, tal que $b_p(\tau\pi, \sigma) \leqslant 2$. No entanto, pelo Lema 1.5, isso só é possível se $\tau\pi = \sigma$. Portanto $d_p(\pi, \sigma) = 1$. ■

Com isso podemos provar o seguinte limite superior para a distância de transposição de prefixo.

Teorema 1.11 (Dias e Meidanis [10]) *Para todo par de permutações π e σ, com $\pi \neq \sigma$, temos $d_p(\pi, \sigma) \leqslant b_p(\pi, \sigma) - 2$.*

Prova: Se $\pi \neq \sigma$, então pelo Lema 1.9 podemos aplicar $b_p(\pi, \sigma) - 3$ transposições de prefixo a π obtendo π' tal que $b_p(\pi', \sigma) = 3$. Pelo Lema 1.10, $d_p(\pi', \sigma) = 1$. Com isso precisamos de no máximo $b_p(\pi, \sigma) - 2$ transposições de prefixo para transformar π em σ. ■

Corolário 1.12 *Para todo par de permutações π e σ, com $\pi \neq \sigma$, temos*

$$d(\pi, \sigma) \leqslant b(\pi, \sigma) + [[\pi(1) = \sigma(1)]] - 2.$$

Prova: Como $d(\pi, \sigma) \leqslant d_p(\pi, \sigma)$, basta aplicar a relação entre b e b_p para obtermos o resultado desejado. ■

1.2.5 Ciclos

Bafna e Pevzner [4] desenvolveram o *diagrama de ciclos* (originalmente chamado de *grafo de ciclos*), uma ferramenta que se mostrou muito útil no estudo de problemas de distância de transposição. Essa estrutura também foi definida como o *diagrama de realidade e desejo* por Meidanis, Walter e Dias [23]. Este diagrama é um grafo onde existe uma ordem entre seus vértices.

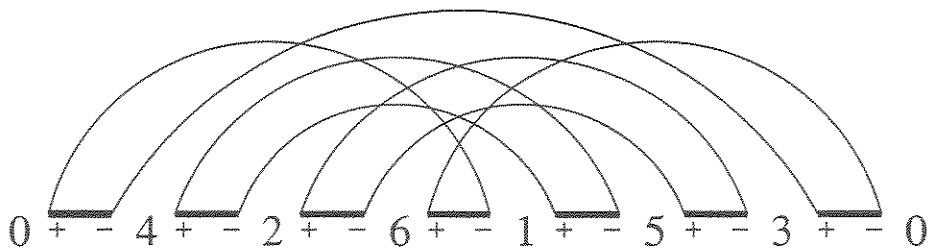

Figura 1.1: Diagrama de ciclos para $\pi = [4,\ 2,\ 6,\ 1,\ 5,\ 3]$.

A seqüência de vértices do diagrama de ciclos é formada da seguinte maneira: para cada elemento $\pi(i)$ da permutação π, crie um par $-\pi(i)$ e $+\pi(i)$, nessa ordem. Adicione um vértice $+0$ no início da seqüência e um vértice -0 no final. As arestas do diagrama são de dois tipos: arestas *desejo* (ou *cinzas*) e arestas *realidade* (ou *pretas*). As arestas realidade são desenhadas ligando os vértices $+\pi(i)$ e $-suc(\pi(i), \pi)$, para $0 \leqslant i \leqslant n$. As arestas desejo são desenhadas ligando vértices $+e$ e $-suc(e)$ para $0 \leqslant e \leqslant n$.

Uma vez definido o diagrama, precisamos agora definir seus ciclos. O tamanho de um ciclo no diagrama de ciclos é definido como o número de arestas realidade que compõem o ciclo. Chamaremos de $c(\pi)$ o número de ciclos no diagrama de ciclos de π. De forma similar, $c_{odd}(\pi)$ será o número de ciclos de tamanho ímpar (ciclos ímpares) no diagrama de ciclos de π. Representaremos por $c_t(\pi)$ o número de ciclos triviais (de tamanho 1) de π.

Sempre que temos um ciclo trivial, existem i e e tais que $\pi(i) = e$ e $suc(\pi(i), \pi) = suc(e)$, ou seja $suc(e, \pi) = suc(e)$ e, portanto, o elemento e não é um ponto de quebra. Por outro lado, se e é um ponto de quebra, não existe um i tal que $\pi(i) = e$ e $suc(\pi(i), \pi) = suc(e)$ e, portanto, e não está em um ciclo trivial. Com essa observação fica fácil perceber que existe exatamente um ciclo trivial para cada elemento que não é ponto de quebra, o que nos dá a seguinte relação.

Proposição 1.13 *Seja π uma permutação de comprimento n. Temos então $b(\pi) = n + 1 - c_t(\pi)$*

A Figura 1.1 mostra um exemplo completo de um diagrama de ciclos. O diagrama é composto por dois ciclos de tamanho 2 e um ciclo de tamanho 3. Note que a única permutação de comprimento n com $n+1$ ciclos é a permutação com todos seus elementos em ordem.

Dada uma permutação π e uma transposição τ, definimos $\Delta c(\pi, \tau)$ como sendo a variação no número de ciclos causado pela aplicação de uma transposição τ, ou seja, $\Delta c(\pi, \tau) = c(\tau\pi) - c(\pi)$. De forma semelhante definimos $\Delta c_{odd}(\pi, \tau)$ para a variação no número de ciclos ímpares de forma que $\Delta c_{odd}(\pi, \tau) = c_{odd}(\tau\pi) - c_{odd}(\pi)$.

Os resultados a seguir foram provados por Bafna e Pevzner [4].

Lema 1.14 *Dada uma permutação π e uma transposição τ, temos*

$$\Delta c_{odd}(\pi, \tau) = \{-2, 0, 2\}.$$

Lema 1.15 *Para toda permutação π, temos $d(\pi) \geqslant \frac{1}{2}(n + 1 - c_{odd}(\pi))$.*

Lema 1.16 *Dada uma permutação não ordenada π de comprimento n, sempre é possível obter ou uma transposição τ tal que $\Delta c_{odd}(\pi, \tau) = 2$ ou três transposições τ_1, τ_2 e τ_3 tais que $\Delta c_{odd}(\pi, \tau_1) = 0$, $\Delta c_{odd}(\tau_1\pi, \tau_2) = 2$ e $\Delta c_{odd}(\tau_2\tau_1\pi, \tau_3) = 2$.*

Lema 1.17 *Para toda permutação π, temos $d(\pi) \leqslant \frac{3}{4}(n + 1 - c_{odd}(\pi))$.*

Prova: Conseqüência direta do Lema 1.16. ■

Teorema 1.18 *Para toda permutação π, temos*

$$\frac{1}{2}(n + 1 - c_{odd}(\pi)) \leqslant d(\pi) \leqslant \frac{3}{4}(n + 1 - c_{odd}(\pi)).$$

Teorema 1.19 *Qualquer algoritmo que produza as transposições como indicadas pelo Lema 1.16 é um algoritmo de aproximação com fator $\frac{3}{2}$ para o problema da distância de transposição.*

Christie [9] obteve um limite inferior mais justo, baseado em um conceito mais complexo que ele define como *barreiras detectáveis*. Se consideramos $h_d(\pi)$ como o número de barreiras detectáveis de uma permutação π, então vale o seguinte teorema.

Teorema 1.20 *Para toda permutação π de comprimento n,*

$$d(\pi) \geqslant \frac{1}{2}(n + 1 - c_{odd}(\pi)) + \left\lceil \frac{\max(h_d(\pi), h_d(\pi^{-1}))}{2} \right\rceil.$$

Este é atualmente o limite inferior mais justo para a distância de transposição. Para maiores informações sobre o conceito de barreiras consulte a tese de doutorado de Christie [9].

Mais tarde Eriksson e colegas [12] apresentaram um algoritmo, sem garantia de aproximação, que ordena qualquer permutação em no máximo $\lfloor \frac{2n-2}{3} \rfloor$ passos e provam que a distância de transposição de uma permutação com seus elementos na ordem inversa à natural é $\lceil \frac{n+1}{2} \rceil$, algo já conhecido por Meidanis e colegas [22] e Christie [9]. Dessa forma obtiveram o seguinte teorema:

Teorema 1.21 *Para $n \geqslant 9$ temos os seguintes limites para o diâmetro de transposição*

$$\frac{n+1}{2} \leqslant D(n) \leqslant \frac{2n-2}{3}.$$

Capítulo 2

Distância de Transposição

2.1 Introdução

Neste capítulo apresentaremos alguns resultados sobre o problema da distância de transposição. Introduziremos a operação de subtração de elementos das permutações e o conceito de subseqüência, idéias pouco usadas na literatura e que ainda não foram totalmente exploradas, mas que mostraram um grande potencial para gerar novos resultados.

Dada uma permutação π dos elementos de um conjunto E, a *subtração* $\pi - a$ de um elemento a é definida como a permutação dos elementos do conjunto $E - \{a\}$, onde a ordem dos seus elementos é a mesma que em π. Note que não exigimos que a seja um elemento de E. A subtração $\pi \setminus S$ de um conjunto S é definida como a permutação dos elementos do conjunto $E - S$, onde a ordem dos seus elementos é a mesma que em π. Se $S = \{s_1, s_2, \ldots, s_k\}$, então $\pi \setminus S = \pi - s_1 - s_2 - \ldots - s_k$.

Exemplo Dada uma permutação $\pi = [a, b, c, d, e, f]$ e um conjunto $S = \{b, d, e\}$, temos $\pi - c = [a, b, d, e, f]$ e $\pi \setminus S = [a, c, f]$.

Uma *subseqüência* de uma permutação π é qualquer permutação que pode ser obtida a partir de π por meio de remoções de seus elementos. Uma subseqüência comum entre duas permutações π e σ é uma permutação que é simultaneamente subseqüência de π e de σ.

Exemplo Seja $\pi = [a, d, f, c, e, b]$ e $\sigma = [a, b, c, d, e, f]$. Então $[a, f, e, b]$ é uma subseqüência de π mas não de σ. A permutação $[a, c, e]$ é uma subseqüência comum entre π e σ.

Utilizando essas idéias apresentaremos como encontrar limites inferior e superior para a distância de transposição.

2.2 Limite Inferior

Lema 2.1 *Para toda permutação π e transposição τ temos*

$$d(\pi - a, \pi\tau - a) \leqslant 1.$$

Prova: Se τ é uma transposição que troca dois blocos que não contêm o elemento a, então podemos aplicar a $\pi - a$ a transposição que troca de posição os mesmos dois blocos, obtendo $d(\pi - a, \pi\tau - a) = 1$. Se um dos blocos envolvidos na transposição é formado somente pelo elemento a, então $\pi\tau - a = \pi - a$ e, portanto, $d(\pi - a, \pi\tau - a) = 0$. No caso restante, quando um dos blocos trocados pela transposição contém o elemento a e algum outro, podemos aplicar uma transposição a $\pi - a$ que troca os mesmos blocos que em π ignorando o elemento a, obtendo $d(\pi - a, \pi\tau - a) = 1$. Portanto, $d(\pi - a, \pi\tau - a) \leqslant 1$. ■

Lema 2.2 *Para todo par de permutações π e σ temos*

$$d(\pi - a, \sigma - a) \leqslant d(\pi, \sigma).$$

Prova: Prova por indução na distância de transposição entre π e σ. O caso base $d(\pi, \sigma) = 0$ é trivial.

Para $d(\pi, \sigma) \geqslant 1$, seja τ uma transposição tal que $d(\pi, \sigma) = d(\pi\tau, \sigma) + 1$. Aplicando a hipótese de indução a $\tau\pi$ e σ, temos $d(\pi\tau - a, \sigma - a) \leqslant d(\pi\tau, \sigma)$.

Pelo Lema 2.1, a hipótese de indução e a definição de τ, concluímos que

$$\begin{aligned} d(\pi - a, \sigma - a) &\leqslant d(\pi - a, \pi\tau - a) + d(\pi\tau - a, \sigma - a) \\ &\leqslant 1 + d(\pi\tau, \sigma) \\ &= d(\pi, \sigma). \end{aligned}$$

■

Note que em ambos os lemas anteriores não é condição necessária que o elemento a faça parte das permutações π ou σ.

Teorema 2.3 *Para todo par de permutações π e σ e conjunto S temos*

$$d(\pi \setminus S, \sigma \setminus S) \leqslant d(\pi, \sigma).$$

Prova: Este teorema é conseqüência direta da aplicação repetida do Lema 2.2. ■

Em outras palavras, o que o Teorema 2.3 quer dizer é que, se estamos ordenando uma permutação, então estamos ordenando todas as suas subseqüências. Dessa forma,

ordenar por transposições qualquer uma dessas subseqüências não pode precisar de mais transposições do que ordenar a permutação original. Este limite inferior é muito fácil de ser calculado e é particularmente útil se a permutação a ser ordenada possui uma grande subseqüência que se sabe que é difícil de ordenar. No entanto, em outros casos este limite inferior pode não acrescentar nenhuma informação nova.

Exemplo Seja $\pi = [7, 1, 6, 5, 4, 3, 2]$. O limite inferior de Bafna e Pevzner [4], para essa permutação é

$$d(\pi) \geqslant \frac{7 + 1 - c_{odd}(\pi)}{2} = \frac{7 + 1 - 2}{2} = 3.$$

No entanto, utilizando o Teorema 2.3 com $S = \{1\}$, obtemos

$$d(\pi) \geqslant d([7, 6, 5, 4, 3, 2]) = d([6, 5, 4, 3, 2, 1]) = \left\lfloor \frac{6}{2} \right\rfloor + 1 = 4,$$

o que é um limite inferior melhor.

2.3 Limite Superior

Lema 2.4 *Para todo par de permutações π e σ, se a é um elemento de π e σ tal que $pred(a, \pi) = pred(a, \sigma)$, então temos*

$$d(\pi - a, \sigma - a) \geqslant d(\pi, \sigma).$$

Prova: Seja $p = pred(a, \pi) = pred(a, \sigma)$. O resultado desejado vem do fato de que toda transposição realizada em $\pi - a$ pode ser reproduzida em π movendo-se os elementos comuns de $\pi - a$ e π da mesma forma e mantendo-se sempre o elemento a logo após o elemento p. ■

Teorema 2.5 *Para todo par de permutações π e σ, se a é um elemento de π e σ tal que $pred(a, \pi) = pred(a, \sigma)$, então temos*

$$d(\pi - a, \sigma - a) = d(\pi, \sigma).$$

Prova: Resultado direto dos Lemas 2.2 e 2.4. ■

Lema 2.6 *Sejam π e σ duas permutações dos elementos de um conjunto E e a o elemento de maior índice em π tal que $pred(a, \pi) = pred(a, \sigma)$. Então, para todo $e \in E \setminus \{a\}$, $pred(e, \pi) = pred(e, \sigma)$ se, e somente se, $pred(e, \pi - a) = pred(e, \sigma - a)$.*

Prova: Claramente a remoção de a só afeta os predecessores de $s = suc(a, \pi)$ e $t = suc(a, \sigma)$. Portanto basta provar que

$$pred(e, \pi) = pred(e, \sigma) \iff pred(e, \pi - a) = pred(e, \sigma - a)$$

para $e \in \{s, t\}$.

Pela forma como escolhemos a, sabemos que

$$pred(s, \sigma) \neq pred(s, \pi) = a = pred(t, \sigma).$$

Portanto $s \neq t$. Como $pred(s, \pi) = a$, então $pred(t, \pi) \neq pred(t, \sigma)$.

Seja $p = pred(a, \pi) = pred(a, \sigma)$. Temos que $pred(s, \pi - a) = p = pred(t, \sigma - a)$. Portanto, $pred(s, \sigma - a) \neq pred(s, \pi - a)$ e $pred(t, \pi - a) \neq pred(t, \sigma - a)$. Com isso, para $e \in \{s, t\}$, temos, necessariamente, $pred(e, \pi) \neq pred(e, \sigma)$ e $pred(e, \pi - a) \neq pred(e, \sigma - a)$, o que confirma a equivalência que queríamos provar, completando a prova de nosso lema. ■

Uma conseqüência do Teorema 2.5 é que não precisamos criar pontos de quebra para transformar otimamente uma permutação em outra, como foi publicado anteriormente por Christie [9, Teorema 3.2.2]:

Teorema 2.7 *Sejam π e σ permutações dos elementos de um conjunto E e A o conjunto dos elementos a de E tais $pred(a, \pi) = pred(a, \sigma)$, ou seja, $A = E \setminus B(\pi, \sigma)$. Temos então*

$$d(\pi, \sigma) = d(\pi \setminus A, \sigma \setminus A).$$

Prova: Prova por indução no tamanho do conjunto subtraído A. O caso base, quando o conjunto A é vazio, é trivial. Consideraremos então que A contém pelo menos um elemento.

Sejam a um elemento do conjunto A com maior índice em π, $\pi' = \pi \setminus (A \setminus \{a\})$ e $\sigma' = \sigma \setminus (A \setminus \{a\})$. Por hipótese de indução, $d(\pi, \sigma) = d(\pi', \sigma')$. Como $pred(a, \pi) = pred(a, \sigma)$, pela aplicação repetida do Lema 2.6 obtemos $pred(a, \pi') = pred(a, \sigma')$, e então utilizamos o Teorema 2.5 para concluir que

$$d(\pi, \sigma) = d(\pi', \sigma') = d(\pi' - a, \sigma' - a) = d(\pi \setminus A, \sigma \setminus A).$$

■

Lema 2.8 *Para todo par de permutações π e σ, temos*

$$d(\pi, \sigma) \leqslant d(\pi - a, \sigma - a) + 1,$$

onde a é um elemento de π e σ.

Prova: Se $pred(a,\pi) = pred(a,\sigma)$, então, pelo Teorema 2.5, temos $d(\pi,\sigma) = d(\pi - a, \sigma - a) \leqslant d(\pi - a, \sigma - a) + 1$.

Se $p = pred(a,\pi)$ é diferente de $q = pred(a,\sigma)$, então podemos aplicar uma transposição τ a π que remove a da posição seguinte a p e insere-o de volta logo após q. Dessa forma, podemos aplicar o Teorema 2.5 a $\pi\tau$ e σ, obtendo $d(\pi\tau,\sigma) = d(\pi\tau - a, \sigma - a) = d(\pi - a, \sigma - a)$, pois $\pi\tau - a = \pi - a$. Com isso concluímos que

$$d(\pi,\sigma) \leqslant d(\pi,\pi\tau) + d(\pi\tau,\sigma) = 1 + d(\pi - a, \sigma - a).$$

■

Observe que se a pertencer a apenas uma das permutações π e σ, temos um valor infinito no lado esquerdo da desigualdade do Lema 2.8, daí a necessidade da hipótese sobre a, pois o lado direito pode ser finito.

Corolário 2.9 *Para todo par de permutações π e σ e todo conjunto S de elementos comuns a π e σ, temos*

$$d(\pi,\sigma) \leqslant d(\pi \setminus S, \sigma \setminus S) + |S|.$$

Prova: Este resultado é conseqüência direta da aplicação repetida do Lema 2.8. ■

Com isso podemos obter o seguinte teorema, anteriormente apresentado por Guyer, Heath e Vergara [16].

Teorema 2.10 *Para todo par de permutações π e σ de comprimento n temos*

$$d(\pi,\sigma) \leqslant n - k,$$

onde k é o tamanho máximo de uma subseqüência comum a π e σ.

Prova: Seja ρ uma subseqüência comum a π e σ de comprimento k e S o conjunto dos elementos de π e σ que não estão em ρ. Então $|S| = n - k$. Ao remover de π e σ os elementos de S, o Corolário 2.9 resulta em

$$\begin{aligned} d(\pi,\sigma) &\leqslant d(\pi \setminus S, \sigma \setminus S) + |S| \\ &= d(\rho,\rho) + n - k \\ &= n - k. \end{aligned}$$

■

Exemplo Seja $\pi_k = [1, 3, 5, \ldots, 2k-1, 2, 4, 6, \ldots, 2k]$. De acordo com o limite superior de Bafna e Pevzner, temos

$$d(\pi_k) \leqslant \frac{3}{4}(2k + 1 - c_{odd}(\pi_k)) = \frac{3}{4}(2k-2) = \frac{3}{2}(k-1),$$

já que $c_{odd}(\pi_k) = 3$ para todo $k \geqslant 1$. Por outro lado, π_k possui uma subseqüência crescente de comprimento $k+1$ e, portanto, pelo Teorema 2.10 concluímos que

$$d(\pi_k) \leqslant 2k - (k+1) = k - 1,$$

um limite superior muito melhor.

O exemplo anterior mostra que o limite superior encontrado pode ficar a uma distância proporcional ao tamanho da permutação mais justo do que o limite superior de Bafna e Pevzner [4]. No entanto há casos em que o limite de Bafna e Pevzner é melhor. Se estabelecermos um limite superior que considera o mínimo de ambos os limites, então nós temos um limite superior melhor que ambos.

2.4 Conclusões e Problemas em Aberto

Apresentamos neste capítulo o uso da subtração de elementos de permutações e a idéia de subseqüência e, com isso, foi possível obter resultados novos. Apesar de tais resultados poderem parecer um pouco específicos em um primeiro momento, eles são apenas demonstrações iniciais de que os conceitos utilizados podem ser bastante úteis.

Podemos a partir daí aprofundar o estudo e tentar generalizar as idéias, como quando consideramos subseqüências circulares e "esqueletos" de ordenações. Esboçaremos essas idéias abaixo.

Informalmente, uma subseqüência circular de uma permutação é uma subseqüência que pode sair pelo fim da permutação e continuar em seu início, contando que não se sobreponha. Formalmente, uma subseqüência circular de comprimento k de uma permutação π de comprimento n é uma seqüência $[\pi(i_1), \pi(i_2), \ldots, \pi(i_{k-1}), \pi(i_k)]$ onde $i_j = (i_1 + d_j) \bmod n$ para todo $1 \leqslant j \leqslant k$ e $0 = d_1 < d_2 < \ldots < d_{k-1} < d_k < n$.

Uma subseqüência circular comum a duas permutações é uma seqüência ρ tal que seja simultaneamente subseqüência circular de ambas as permutações. Na verdade tal subseqüência circular comum só precisa ser circular em uma das permutações, já que podemos redefinir o início da subseqüência ρ de forma que ela seja uma subseqüência normal de uma das permutações.

Exemplo Considere as permutações $\pi = [f, d, h, a, b, g, c, e]$ e $\sigma = [d, e, g, f, h, a, c, b]$. Temos que a seqüência $\rho = [a, b, e, f, h]$ é subseqüência circular tanto de π quanto de

σ. No entanto podemos deslocar circularmente os elementos de ρ duas posições para a esquerda, obtendo a seqüência $s' = [e, f, h, a, b]$, que também é uma subseqüência circular comum à π e σ, sendo que com relação à σ é uma subseqüência normal. Repare que a permutação obtida pelo deslocamento circular dos elementos de π uma posição para a direita também possui a seqüência s' como subseqüência.

É importante notar que se temos uma subseqüência circular ρ de uma permutação π então sempre podemos deslocar circularmente os elementos de π de forma que a nova permutação possua ρ como uma subseqüência normal. Além disso, todo deslocamento circular de d posições para a esquerda dos elementos de uma permutação π de comprimento n nada mais é do que a aplicação da transposição de prefixo $\tau(1, 1+d, n+1)$ à permutação π.

Com isso podemos aperfeiçoar o limite superior do Teorema 2.10 considerando também subseqüências circulares. Seja ρ uma subseqüência circular de tamanho k comum às permutações π e σ. Então existe uma seqüência ρ' também de comprimento k e uma transposição τ tal que ρ' seja uma subseqüência comum a $\pi\tau$ e σ. Portanto

$$d(\pi, \sigma) \leqslant d(\pi\tau, \sigma) + 1 \leqslant n - k + 1.$$

Com isso fica claro que conseguiremos um limite mais justo sempre que a maior subseqüência circular comum a ambas as permutações for 2 unidades maior que a maior subseqüência comum.

Na verdade obteremos um ganho nesse limite sempre que, aplicando uma transposição à uma das permutações, o comprimento da maior subseqüência comum a ambas aumentar em pelo menos duas unidades. Portanto é interessante tentar descobrir como aplicar transposições que propiciem uma maior subseqüência comum, possibilitando esse ganho.

Considere uma *transformação* como a seqüência de permutações durante os passos de transformação de uma permutação π em outra σ. Chamaremos de π_i a permutação obtida após a aplicação das i primeiras transposições à permutação π. Neste caso, sendo k a distância entre π e σ, temos que uma transformação ótima é composta pelas permutações $\pi_0 = \pi, \pi_1, \pi_2, \ldots, \pi_{k-1}, \pi_k = \sigma$.

Exemplo Para transformar uma permutação $\pi = [c, b, e, a, d]$ em $\sigma = [a, b, c, d, e]$ temos

$$\begin{aligned} \pi_0 &= [c, b, e, a, d] \\ \pi_1 &= [b, c, e, a, d] \\ \pi_2 &= [a, b, c, e, d] \\ \pi_3 &= [a, b, c, d, e]. \end{aligned}$$

Se temos uma subseqüência e_0 comum a todas as permutações π_i isso quer dizer que os elementos de e_0 nunca mudam de ordem durante a transformação e portanto, nenhuma das transposições pode conter elementos de e_0 em ambos os blocos que ela troca de lugar. Perceba que isso pode limitar bastante as transposições. No entanto, se a transformação é longa, a tendência é que tal subseqüência comum e_0 seja muito pequena, podendo até ter tamanho um, o que não ajuda em nada. Podemos então olhar para subseqüências comuns a permutações mais adiantes na transformação.

Chamaremos de $e_i, i > 0$ uma subseqüência comum a todas as permutações π_j com $i \leqslant j \leqslant k$ e que tenha e_{i-1} como subseqüência. Isso força que o comprimento de e_i seja sempre maior ou igual ao comprimento de e_{i-1}.

Exemplo Na transformação do exemplo anterior temos as seguintes possibilidades maximais para os diversos e_i.

$$\begin{aligned} e_0 &= [a,d]\ [b,d]\ [b,e]\ [c,d]\ [c,e] \\ e_1 &= [a,d]\ [b,c,d]\ [b,c,e] \\ e_2 &= [a,b,c,d]\ [a,b,c,e] \\ e_3 &= [a,b,c,d,e]. \end{aligned}$$

Dessa forma, os diversos e_i sugerem uma estrutura para a transformação. Seus elementos dividem a permutação π_i em intervalos. Os elementos contidos estritamente em cada um desses intervalos devem obrigatoriamente sair de seu intervalo em algum momento da transformação, pois caso contrário tal elemento poderia fazer parte de e_i. É importante notar que em geral, além das diversas possibilidades para os e_i, ainda podemos realizar a transformação de uma permutação em outra de várias formas.

Na situação apresentada, estamos obtendo os e_i a partir de uma ordenação. Uma questão interessante é se é possível encontrar um e_{i+1} válido a partir de e_i e da permutação de origem e de destino. Se isso for possível em tempo polinomial então também conseguiremos transformar permutações otimamente em tempo polinomial.

Ainda não está claro como essas idéias poderiam contribuir de forma mais significante no cálculo da distância de transposição, que continua em aberto, mas os resultados deste capítulo indicam que explorar mais extensivamente essas outras formas de ver o problema da distância de transposição podem trazer bons resultados.

Capítulo 3

Distância de Transposição de Prefixo

3.1 Introdução

Diante das dificuldades encontradas no problema de distância de transposição, Dias e Meidanis [10] propuseram a variação do problema que considera apenas transposições prefixo para transformar uma permutação em outra. É importante notar que resolver o problema para transposições de prefixo não implica necessariamente em resolver o problema original. No entanto acreditamos que o estudo desta variação possa resultar em um maior conhecimento do problema original. Além disso, a natureza de ambos os problemas é bem próxima, o que pode possibilitar a aplicação das técnicas bem sucedidas em um, no outro.

Neste capítulo, tentaremos nos aproximar da resolução do problema de distância de transposição de prefixo de duas formas. A primeira é estudando a permutação onde os elementos aparecem na ordem invertida, também chamada de permutação reversa. Acredita-se que essa esteja entre as permutações mais difíceis de se ordenar. Descobrir por que ela é difícil de se ordenar pode nos dar novas importantes informações.

A segunda abordagem do problema é o inverso. Analisaremos justamente as permutações que são fáceis de se ordenar, na busca de se caracterizar os fatores que tornam a ordenação dessas permutações fácil.

3.2 Ordenação da Permutação Reversa

A permutação reversa é aquela que apresenta seus elementos na ordem inversa à natural. Chamaremos a permutação reversa com comprimento n de R_n. Se a permutação for dos primeiros números inteiros positivos, então $R_n = [n, n-1, n-2, \ldots, 3, 2, 1]$.

Conjectura-se [10] que a permutação reversa esteja entre a mais difíceis de se orde-

nar, de forma que sua distância de transposição de prefixo seja igual ao diâmetro de transposição de prefixo ($d_p(R_n) = D(n)$).

O estudo de tal permutação é uma tentativa de se encontrar um limite superior para a distância de transposição de prefixo e as propriedades que tornam uma ordenação difícil. Conhecendo tais propriedades, poderíamos aprender como contorná-las para obter uma ordenação ótima ou que se aproxime bastante do ótimo.

Os resultados desta seção já foram apresentads anteriomente em um relatório técnico [13].

3.2.1 Algoritmo para Ordenar a Permutação Reversa

Em seu trabalho pioneiro, Dias e Meidanis [10] apresentam, sem prova, um algoritmo para ordenar por transposições de prefixo a permutação reversa R_n. Inspirados nesse algoritmo, propomos um novo algoritmo (Algoritmo 1) para ordenar R_n, onde apresentamos explicitamente quais as transposições a serem realizadas e o formato da permutação em cada etapa e, o mais importante, uma prova formal de sua corretude.

A Figura 3.1 mostra um exemplo de execução do algoritmo para a ordenação de R_{16}.

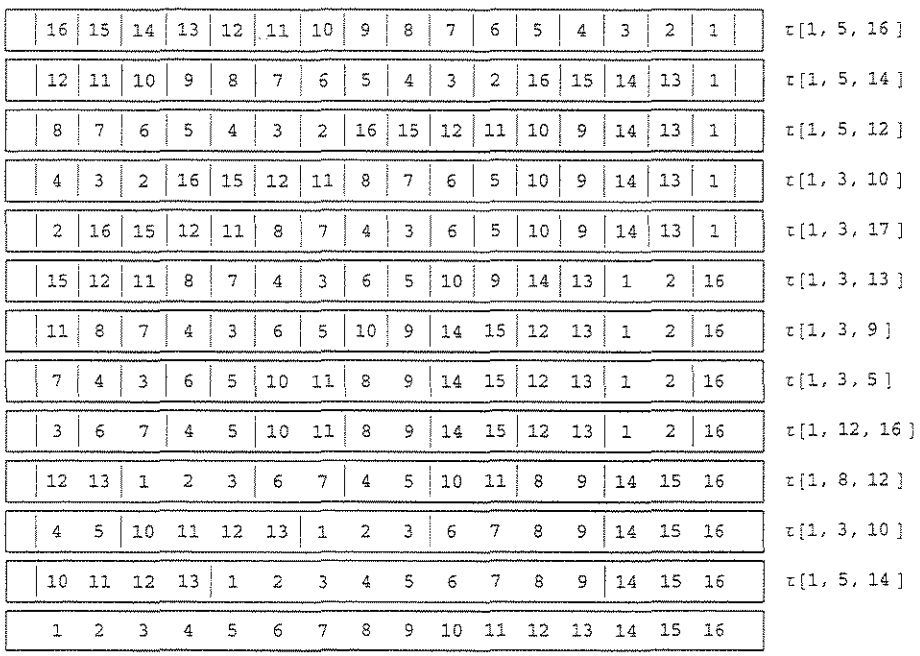

Figura 3.1: Exemplo de execução do algoritmo de ordenação da permutação reversa. Na figura estão as permutações ao longo da ordenação acompanhadas das transposições realizadas. As barras verticais indicam pontos de quebra.

A prova de corretude do Algoritmo 1 será realizada dividindo-o em diversas fases e provando a transição entre elas. Isso é feito por meio da caracterização da permutação

Algoritmo 1 Algoritmo para ordenar R_n.

Require: $\pi = R_n$, com $n \geqslant 4$
Ensure: π ordenado

▷ Fase 0: Reduz para n múltiplo de 4
for $i \leftarrow 0$ **to** $n \bmod 4 - 1$ **do**
  $\pi \leftarrow \pi\tau(1, 2, n + 1 - i)$
$n \leftarrow n - (n \bmod 4)$

▷ Fase 1 – Neste ponto $\pi = R_n$, onde n é múltiplo de 4
for $i \leftarrow 0$ **to** $n/4 - 2$ **do**
  $\pi \leftarrow \pi\tau(1, 5, n - 2i)$
$\pi \leftarrow \pi\tau(1, 3, n/2 + 2)$

▷ Fase 2 – Neste ponto $\pi = F_1(n)$
for $i \leftarrow 0$ **to** $n/4 - 1$ **do**
  $\pi \leftarrow \pi\tau(1, 3, n + 1 - 4i)$

▷ Fase 3 – Neste ponto $\pi = F_2(n)$
for $i \leftarrow 0$ **to** $\lfloor n/8 \rfloor - 1$ **do**
  $\pi \leftarrow \pi\tau(1, n - 4 - 4i, n - 4i)$
if $n \bmod 8 = 0$ **then**
  $\pi \leftarrow \pi\tau(1, 3, n/2 + 2)$
else
  $\pi \leftarrow \pi\tau(1, n/2, n/2 + 2)$

▷ Fase 4 – Neste ponto $\pi = F_3(n)$
for $i \leftarrow 0$ **to** $\lceil n/8 \rceil - 2$ **do**
  $\pi \leftarrow \pi\tau(1, 5, 4\lfloor n/8 \rfloor + 6 + 4i)$
▷ Neste ponto $\pi = \iota_n$

em cada passo do processo de ordenação.

A *Fase 0* é responsável por reduzir o problema de ordenar R_n com n qualquer para o problema de ordenar R_n com n múltiplo de 4. O que essa fase faz nada mais é do que posicionar os $n \bmod 4$ elementos extras, um de cada vez, em suas posições finais no extremo final da permutação. Dessa forma esses elementos podem ser deixados de lado deste ponto em diante e assim consideraremos n como sendo múltiplo de 4 no restante do algoritmo.

Exemplo Para $n = 6$, a Fase 0 realiza a seguinte transformação:

$$[6,\ 5,\ 4,\ 3,\ 2,\ 1] \rightarrow [4,\ 3,\ 2,\ 1,\ 5,\ 6].$$

Após essa transformação o problema se reduz a ordenar R_n com $n = 4$.

Para caracterizar a permutação após cada fase intermediária do algoritmo, nós a trataremos como uma seqüência e utilizaremos um operador de concatenação representado pelo símbolo $\odot$. Dessa forma a seqüência $a \odot b$ é a concatenação da seqüência a com a seqüência b.

Também definiremos um operador de concatenação generalizado tal que $\bigodot_{j=a}^{b} f(j)$ é a seqüência resultante da concatenação das seqüências $f(j)$, com j variando de a a b.

Exemplo As seguintes igualdades ilustram o uso dos operadores:

$$[1,2,3] \odot [4,5,6] = [1,2,3,4,5,6],$$

$$\bigodot_{j=0}^{4} [1+3j, 2+3j] = [1,2,4,5,7,8,10,11,13,14].$$

Definiremos agora algumas famílias de permutações que chamaremos de F_1, F_2 e F_3. Veremos mais tarde que elas representam as permutações resultantes das fases 1, 2 e 3 do Algoritmo 1, respectivamente. As definições fazem sentido apenas para n múltiplo de 4.

Definição A permutação $F_1(n)$ é definida pela seguinte fórmula:

$$F_1(n) = [2] \odot \bigodot_{j=0}^{n/4-2} [n-4j,\ n-1-4j] \odot [4,\ 3] \odot \bigodot_{j=0}^{n/4-2} [6+4j,\ 5+4j] \odot [1].$$

Exemplo

$$F_1(12) = [2,\ 12,\ 11,\ 8,\ 7,\ 4,\ 3,\ 6,\ 5,\ 10,\ 9,\ 1]$$

Definição A permutação $F_2(n)$ é definida pela seguinte fórmula:

$$F_2(n) = [3] \odot \bigodot_{j=0}^{n/4-2} [6+4j,\ 7+4j,\ 4+4j,\ 5+4j] \odot [1,\ 2,\ n].$$

Exemplo

$$F_2(12) = [3,\ 6,\ 7,\ 4,\ 5,\ 10,\ 11,\ 8,\ 9,\ 1,\ 2,\ 12]$$

Definição A permutação $F_3(n)$ é definida pela seguinte fórmula:

$$\begin{aligned} F_3(n) \ = \ & \bigodot_{j=0}^{\lceil n/8 \rceil - 2} \bigodot_{k=0}^{3} [10 - n \bmod 8 + 8j + k] \odot [1,\ 2,\ 3,\ 4,\ 5] \odot \\ & \bigodot_{j=0}^{\lfloor n/8 \rfloor - 2} \bigodot_{k=0}^{3} [6 + n \bmod 8 + 8j + k] \odot [n-2,\ n-1,\ n]. \end{aligned}$$

Exemplo

$$F_3(16) = [10,\ 11,\ 12,\ 13,\ 1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 14,\ 15,\ 16],$$

$$F_3(20) = [6,\ 7,\ 8,\ 9,\ 14,\ 15,\ 16,\ 17,\ 1,\ 2,\ 3,\ 4,\ 5,\ 10,\ 11,\ 12,\ 13,\ 18,\ 19,\ 20].$$

Provaremos que o Algoritmo 1 de fato ordena a permutação reversa R_n provando que a *Fase 1* transforma R_n em $F_1(n)$, a *Fase 2* transforma $F_1(n)$ em $F_2(n)$ e a *Fase 3* transforma $F_2(n)$ na permutação identidade, terminando a ordenação. Vale relembrar que estamos considerando apenas valoers de n múltiplos de 4.

Lema 3.1 *A* Fase 1 *transforma R_n, para $n \geqslant 4$, na permutação $F_1(n)$.*

Prova: Considere π_i, onde $0 \leqslant i \leqslant n/4$, como sendo a permutação obtida após i transposições da *Fase 1*. Então temos a seguinte recorrência:

$$\begin{aligned} \pi_0 &= R_n \\ \pi_i &= \pi_{i-1}\tau(1, 5, n-2i)\ \text{, para } 1 \leqslant i \leqslant n/4 - 1 \\ \pi_{n/4} &= \pi_{n/4-1}\tau(1, 3, n/2+2). \end{aligned}$$

Para $0 \leqslant i \leqslant n/4 - 1$, temos a seguinte fórmula para a permutação π_i:

$$\pi_i = \bigodot_{j=0}^{n-5-4i} [n-4i-j] \odot [4,\ 3,\ 2] \odot \bigodot_{j=0}^{i-1} [n-4j,\ n-1-4j] \odot \bigodot_{j=0}^{i-1} [n+2-4i+4j,\ n+1-4i+4j] \odot [1].$$

Ou, de forma equivalente:

$$\pi_i = \gamma_i \odot [4,\ 3,\ 2] \odot \alpha_i \odot \beta_i \odot [1],$$

onde

$$\begin{aligned} \gamma_i &= \bigodot_{j=0}^{n-5-4i} [n-4i-j] \\ \alpha_i &= \bigodot_{j=0}^{i-1} [n-4j,\ n-1-4j] \\ \beta_i &= \bigodot_{j=0}^{i-1} [n+2-4i+4j,\ n+1-4i+4j]. \end{aligned}$$

A prova dessa afirmação pode ser feita por indução. Para o caso base $i = 0$ nós temos:

$$\begin{aligned}
\pi_0 &= \bigodot_{j=0}^{n-5}[n-j] \odot [4,\ 3,\ 2] \odot \bigodot_{j=0}^{-1}[n-4j,\ n-1-4j] \odot \\
&\quad \bigodot_{j=0}^{-1}[n+2+4j,\ n+1+4j] \odot [1] \\
&= \bigodot_{j=0}^{n-5}[n-j] \odot [4,\ 3,\ 2] \odot [1] \\
&= R_n.
\end{aligned}$$

Vamos provar agora que, para $1 \leqslant i+1 \leqslant n/4-1$, temos $\pi_i\tau(1,5,n-2i) = \pi_{i+1}$. Perceba que o que a i-ésima transposição da *Fase 1* faz é remover um prefixo de γ_i e inserí-lo de volta entre α_i e β_i. Essa observação torna mais claras as seguintes fórmulas relacionando as subseqüências α, β e γ de π_i e π_{i+1}:

$$\begin{aligned}
\gamma_i &= \bigodot_{j=0}^{n-5-4i}[n-4i-j] \\
&= [n-4i,\ n-1-4i,\ n-2-4i,\ n-3-4i] \odot \\
&\quad \bigodot_{j=4}^{n-5-4i}[n-4i-j] \\
&= [n-4i,\ n-1-4i,\ n-2-4i,\ n-3-4i] \odot \\
&\quad \bigodot_{j=0}^{n-5-4(i+1)}[n-4(i+1)-j,\ n-1-4(i+1)-j] \\
&= [n-4i,\ n-1-4i,\ n-2-4i,\ n-3-4i] \odot \gamma_{i+1},
\end{aligned}$$

$$\begin{aligned}
\alpha_{i+1} &= \bigodot_{j=0}^{i}[n-4j,\ n-1-4j] \\
&= \bigodot_{j=0}^{i-1}[n-4j,\ n-1-4j] \odot [n-4i,\ n-1-4i] \\
&= \alpha_i \odot [n-4i,\ n-1-4i],
\end{aligned}$$

$$\begin{aligned}
\beta_{i+1} &= \bigodot_{j=0}^{i}[n-2-4i+4j,\ n-1-4i+4j] \\
&= [n-2-4i,\ n-3-4i] \odot \bigodot_{j=1}^{i}[n-2-4i+4j,\ n-1-4i+4j] \\
&= [n-2-4i,\ n-3-4i] \odot \bigodot_{j=0}^{i-1}[n-2-4(i-1)+4j,\ n-3-4(i-1)+4j] \\
&= [n-2-4i,\ n-3-4i] \odot \beta_i.
\end{aligned}$$

Surge que

$$\begin{aligned}
\pi_i &= \gamma_i \odot [4,\ 3,\ 2] \odot \alpha_i \odot \beta_i \odot [1] \\
&= [n-4i,\ n-1-4i,\ n-2-4i,\ n-3-4i] \odot \gamma_{i+1} \odot [4,\ 3,\ 2] \odot \alpha_i \odot \beta_i \odot [1].
\end{aligned}$$

Perceba que a transposição $\tau(1, 5, n-2i)$ remove os quatro primeiros elementos de π e insere-os de volta entre os blocos α_i e β_i, já que o comprimento de $\beta_i \odot [1]$ é exatamente $2i+1$. Então temos

$$\begin{aligned}
\pi_i\tau(1,5,n-2i) &= \gamma_{i+1} \odot [4,\ 3,\ 2] \odot \alpha_i \odot [n-4i,\ n-1-4i] \odot \\
&\quad [n-2-4i,\ n-3-4i] \odot \beta_i \odot [1] \\
&= \gamma_{i+1} \odot [4,\ 3,\ 2] \odot \alpha_{i+1} \odot \beta_{i+1} \odot [1] \\
&= \pi_{i+1},
\end{aligned}$$

o que prova que após $n/4-1$ transposições da *Fase 1* nós obtemos a permutação

$$\begin{aligned}
\pi_{n/4-1} &= \bigodot_{j=0}^{-1}[4-j] \odot [4,\ 3,\ 2] \odot \bigodot_{j=0}^{n/4-2}[n-4j,\ n-1-4j] \odot \bigodot_{j=0}^{n/4-2}[6+4j,\ 5+4j] \odot [1] \\
&= [4,\ 3,\ 2] \odot \bigodot_{j=0}^{n/4-2}[n-4j,\ n-1-4j] \odot \bigodot_{j=0}^{n/4-2}[6+4j,\ 5+4j] \odot [1].
\end{aligned}$$

A última transposição da *Fase 1* é $\tau(1, 3, n/2+2)$. Então temos

$$\begin{aligned}
\pi_{n/4} &= \pi_{n/4-1}\tau(1,3,n/2+2) \\
&= [2] \odot \bigodot_{j=0}^{n/4-2}[n-4j,\ n-1-4j] \odot [4,\ 3] \odot \bigodot_{j=0}^{n/4-2}[6+4j,\ 5+4j] \odot [1] \\
&= F_1(n),
\end{aligned}$$

concluindo nossa prova. ■

Lema 3.2 *A* Fase 2 *transforma* $F_1(n)$, *para* $n \geqslant 4$, *na permutação* $F_2(n)$.

Prova: Considere π_i, onde $0 \leqslant i \leqslant n/4$, como sendo a permutação obtida após i transposições da *Fase 2*. Então temos a seguinte recorrência:

$$\begin{aligned} \pi_0 &= F_1(n) \\ \pi_{i+1} &= \pi_i \tau(1, 3, n+1-4i) \text{ , para } 0 \leqslant i \leqslant n/4-1. \end{aligned}$$

Então temos:

$$\begin{aligned} \pi_0 &= F_1(n) \\ &= [2] \odot \bigodot_{j=0}^{n/4-2} [n-4j,\ n-1-4j] \odot [4,\ 3] \odot \bigodot_{j=0}^{n/4-2} [6+4j,\ 5+4j] \odot [1] \\ &= [2,\ n] \odot [n-1] \odot \bigodot_{j=1}^{n/4-1} [n-4j,\ n-1-4j] \odot \bigodot_{j=0}^{n/4-2} [6+4j,\ 5+4j] \odot [1] \\ &= [2,\ n] \odot [n-1] \odot \bigodot_{j=0}^{n/4-2} [n-4-4j,\ n-5-4j] \odot \bigodot_{j=0}^{n/4-2} [6+4j,\ 5+4j] \odot [1]. \end{aligned}$$

Para $0 \leqslant i \leqslant n/4-1$, temos a seguinte fórmula para a permutação π_{i+1}:

$$\pi_{i+1} = [n-1-4i] \odot \alpha_{i+1} \odot \beta_{i+1} \odot \gamma_{i+1} \odot [1,\ 2,\ n],$$

onde

$$\begin{aligned} \alpha_{i+1} &= \bigodot_{j=0}^{n/4-2-i} [n-4-4i-4j,\ n-5-4i-4j] \\ \beta_{i+1} &= \bigodot_{j=0}^{n/4-2-i} [6+4j,\ 5+4j] \\ \gamma_{i+1} &= \bigodot_{j=0}^{i-1} [n+2-4i+4j,\ n+3-4i+4j,\ n-4i+4j,\ n+1-4i+4j]. \end{aligned}$$

A prova dessa afirmação pode ser feita por indução. Para o caso base $i = 0$, temos

$$\begin{aligned} \pi_1 &= \pi_0 \tau(1, 3, n+1) \\ &= [n-1] \odot \bigodot_{j=0}^{n/4-2} [n-4-4j,\ n-5-4j] \odot \bigodot_{j=0}^{n/4-2} [6+4j,\ 5+4j] \odot [1] \odot [2,\ n] \\ &= [n+3-4 \cdot 1] \odot \alpha_1 \odot \beta_1 \odot \gamma_1 \odot [1,\ 2,\ n], \end{aligned}$$

onde

$$\begin{aligned}
\alpha_1 &= \bigodot_{j=0}^{n/4-2} [n-4-4j,\ n-5-4j] \\
\beta_1 &= \bigodot_{j=0}^{n/4-2} [6+4j,\ 5+4j] \\
\gamma_1 &= \bigodot_{j=0}^{-1} [n+2+4j,\ n+3+4j,\ n+4j,\ n+1+4j] = [\,].
\end{aligned}$$

Vamos provar agora que, para $1 \leqslant i \leqslant n/4-1$, temos $\pi_i\tau(1,3,n+1-4i) = \pi_{i+1}$. É importante verificar as seguintes relações entre subseqüências de π_i e π_{i+1}:

$$\begin{aligned}
\alpha_i &= \bigodot_{j=0}^{n/4-2-(i-1)} [n-4i-4j,\ n-4i-4j-1] \\
&= [n-4i,\ n-1-4i] \odot \bigodot_{j=1}^{n/4-1-i} [n-4i-4j,\ n-1-4i-4j] \\
&= [n-4i,\ n-1-4i] \odot \bigodot_{j=0}^{n/4-2-i} [n-4(i+1)-4j,\ n-1-4(i+1)-4j] \\
&= [n-4i,\ n-1-4i] \odot \alpha_{i+1}
\end{aligned}$$

$$\begin{aligned}
\beta_i &= \bigodot_{j=0}^{n/4-1-i} [6+4j,\ 5+4j] \\
&= \bigodot_{j=0}^{n/4-1-(i+1)} [6+4j,\ 5+4j] \odot [n+2-4i,\ n+1-4i] \\
&= \beta_{i+1} \odot [n+2-4i,\ n+1-4i]
\end{aligned}$$

$$\begin{aligned}
\gamma_{i+1} &= \bigodot_{j=0}^{i-1} [n+2-4i+4j,\ n+3-4i+4j,\ n-4i+4j,\ n+1-4i+4j] \\
&= [n+2-4i,\ n+3-4i,\ n-4i,\ n+1-4i] \odot \\
&\quad \bigodot_{j=0}^{i-2} [n+6-4i+4j,\ n+7-4i+4j,\ n+4-4i+4j,\ n+5-4i+4j] \\
&= [n+2-4i,\ n+3-4i,\ n-4i,\ n+1-4i] \odot \gamma_i.
\end{aligned}$$

BIBLIOTECA CENTRAL
DESENVOLVIMENTO
UNICAMP

Surge que

$$\begin{aligned}
\pi_i &= [n+3-4i] \odot \alpha_i \odot \beta_i \odot \gamma_i \odot [1,\ 2,\ n] \\
&= [n+3-4i] \odot [n-4i,\ n+1-4i] \odot \alpha_{i+1} \odot \\
&\quad \beta_{i+1} \odot [n+2-4i,\ n+1-4i] \odot \gamma_i \odot [1,\ 2,\ n] \\
&= [n+3-4i,\ n-4i] \odot [n-1-4i] \odot \alpha_{i+1} \odot \\
&\quad \beta_{i+1} \odot [n+2-4i] \odot [n+1-4i] \odot \gamma_i \odot [1,\ 2,\ n].
\end{aligned}$$

Perceba que a transposição $\tau(1,3,n+1-4i)$ remove o prefixo $[n+3-4i,\ n-4i]$ de π_i e insere-o de volta entre os blocos $[n+2-4i]$ e $[n+1-4i]$, já que o comprimento do sufixo $[n+1-4i] \odot \gamma_i \odot [1,\ 2,\ n]$ é exatamente $1+4(i-1)+3 = 4i$. Temos então

$$\begin{aligned}
\pi_{i+1} &= \pi_i \tau(1,3,n+1-4i) \\
&= [n-4i-1] \odot \alpha_{i+1} \odot \beta_{i+1} \odot \\
&\quad [n-4i+2] \odot [n+3-4i,\ n-4i] \odot [n+1-4i] \odot \gamma_i \odot [1,\ 2,\ n] \\
&= [n-1-4i] \odot \alpha_{i+1} \odot \beta_{i+1} \odot \\
&\quad [n+2-4i,\ n+3-4i,\ n-4i,\ n+1-4i] \odot \gamma_i \odot [1,\ 2,\ n] \\
&= [n-1-4i] \odot \alpha_{i+1} \odot \beta_{i+1} \odot \gamma_{i+1} \odot [1,\ 2,\ n],
\end{aligned}$$

o que prova que após todas as $n/4$ transposições da *Fase 2* nós obtemos a permutação

$$\begin{aligned}
\pi_{n/4} &= [3] \odot \alpha_{n/4} \odot \beta_{n/4} \odot \gamma_{n/4} \odot [1,\ 2,\ n] \\
\alpha_{n/4} &= \bigodot_{j=0}^{-1} [-4j,\ -1-4j] = [\,] \\
\beta_{n/4} &= \bigodot_{j=0}^{-1} [6+4j,\ 5+4j] = [\,] \\
\gamma_{n/4} &= \bigodot_{j=0}^{n/4-2} [6+4j,\ 7+4j,\ 4+4j,\ 5+4j].
\end{aligned}$$

Isto é,

$$\begin{aligned}
\pi_{n/4} &= [3] \odot \bigodot_{j=0}^{n/4-2} [6+4j,\ 7+4j,\ 4+4j,\ 5+4j] \odot [1,\ 2,\ n] \\
&= F_2(n),
\end{aligned}$$

concluindo nossa prova. ■

Lema 3.3 *A* Fase 3 *transforma* $F_2(n)$, *para* $n \geqslant 8$, *na permutação* $F_3(n)$.

Prova: Considere π_i, onde $0 \leqslant i \leqslant \lfloor n/8 \rfloor + 1$, como sendo a permutação obtida após i transposições da *Fase 3*. Então temos a seguinte recorrência:

$$\begin{aligned}
\pi_0 &= F_2(n) \\
\pi_{i+1} &= \pi_i \tau(1, n-4-4i, n-4i) \text{ , para } 0 \leqslant i \leqslant \lfloor n/8 \rfloor - 1 \\
\pi_{\lfloor n/8 \rfloor + 1} &= \begin{cases} \pi_{\lfloor n/8 \rfloor} \tau(1, 3, n/2+2) & \text{, se } n \bmod 8 = 0 \\ \pi_{\lfloor n/8 \rfloor} \tau(1, n/2, n/2+2) & \text{, se } n \bmod 8 \neq 0. \end{cases}
\end{aligned}$$

Então temos, para $n \geqslant 8$:

$$\begin{aligned}
\pi_0 &= F_2(n) \\
&= [3] \odot \bigodot_{j=0}^{n/4-2} [6+4j,\ 7+4j,\ 4+4j,\ 5+4j] \odot [1,\ 2,\ n] \\
&= [3] \odot \bigodot_{j=0}^{n/4-3} [6+4j,\ 7+4j,\ 4+4j,\ 5+4j] \odot \\
&\quad [n-2,\ n-1,\ n-4,\ n-3,\ 1,\ 2,\ n].
\end{aligned}$$

Para $0 \leqslant i \leqslant \lfloor n/8 \rfloor - 1$, temos, para a permutação π_{i+1}, a seguinte formula:

$$\pi_{i+1} = [n-4-8i,\ n-3-8i] \odot \alpha_{i+1} \odot [1,\ 2,\ 3] \odot \beta_{i+1} \odot \gamma_{i+1} \odot [n-2,\ n-1,\ n],$$

onde

$$\begin{aligned}
\alpha_{i+1} &= \bigodot_{j=0}^{i-1} \bigodot_{k=0}^{3} [n+2-8i+8j+k] \\
\beta_{i+1} &= \bigodot_{j=0}^{n/4-3-2i} [6+4j,\ 7+4j,\ 4+4j,\ 5+4j] \\
\gamma_{i+1} &= \bigodot_{j=0}^{i-1} \bigodot_{k=0}^{3} [n-2-8i+8j+k].
\end{aligned}$$

Podemos provar essa afirmação por indução. Para o caso base $i = 0$, temos

$$\begin{aligned}
\pi_1 &= \pi_0 \tau(1, n-4, n) \\
&= [n-4,\ n-3,\ 1,\ 2,\ 3] \odot \\
&\quad \bigodot_{j=0}^{n/4-3} [6+4j,\ 7+4j,\ 4+4j,\ 5+4j] \odot [n-2,\ n-1,\ n] \\
&= [n+4-8,\ n+5-8] \odot \alpha_1 \odot [1,\ 2,\ 3] \odot \beta_1 \odot \gamma_1 \odot [n-2,\ n-1,\ n],
\end{aligned}$$

onde

$$\begin{aligned}
\alpha_1 &= \bigodot_{j=0}^{-1}\bigodot_{k=0}^{3}[n+10-8+8j+k] = [\,]\\
\beta_1 &= \bigodot_{j=0}^{n/4-3}[6+4j,\ 7+4j,\ 4+4j,\ 5+4j]\\
\gamma_1 &= \bigodot_{j=0}^{-1}\bigodot_{k=0}^{3}[n+6-8+8j+k] = [\,].
\end{aligned}$$

Vamos provar agora que, para $1 \leqslant i \leqslant \lfloor n/8 \rfloor - 1$, temos $\pi_i\tau(1, n-4-4i, n-4i) = \pi_{i+1}$. É importante verificar as seguintes relações entre subseqüências de π_i e π_{i+1}:

$$\begin{aligned}
\alpha_{i+1} &= \bigodot_{j=0}^{i-1}\bigodot_{k=0}^{3}[n+2-8i+8j+k]\\
&= \bigodot_{k=0}^{3}[n+2-8i+k] \odot \bigodot_{j=0}^{i-2}\bigodot_{k=0}^{3}[n+2-8(i-1)+8j+k]\\
&= [n+2-8i,\ n+3-8i,\ n+4-8i,\ n+5-8i] \odot \alpha_i
\end{aligned}$$

$$\begin{aligned}
\beta_i &= \bigodot_{j=0}^{n/4-2i-1}[6+4j,\ 7+4j,\ 4+4j,\ 5+4j]\\
&= \bigodot_{j=0}^{n/4-2i-3}[6+4j,\ 7+4j,\ 4+4j,\ 5+4j] \odot\\
&\quad [n-2-8i,\ n-1-8i,\ n-4-8i,\ n-3-8i] \odot\\
&\quad [n+2-8i,\ n+3-8i,\ n-8i,\ n+1-8i]\\
&= \beta_{i+1} \odot [n-2-8i,\ n-1-8i] \odot\\
&\quad [n-4-8i,\ n-3-8i,\ n+2-8i,\ n+3-8i] \odot [n-8i,\ n+1-8i]
\end{aligned}$$

$$\begin{aligned}
\gamma_{i+1} &= \bigodot_{j=0}^{i-1}\bigodot_{k=0}^{3}[n-2-8i+8j+k]\\
&= \bigodot_{k=0}^{3}[n-2-8i+k] \odot \bigodot_{j=0}^{i-2}\bigodot_{k=0}^{3}[n-2-8(i-1)+8j+k]\\
&= [n-2-8i,\ n-1-8i,\ n-8i,\ n+1-8i] \odot \gamma_i.
\end{aligned}$$

Surge que

$$\begin{aligned}\pi_i &= [n+4-8i,\ n+5-8i] \odot \alpha_i \odot [1,\ 2,\ 3] \odot \beta_i \odot \gamma_i \odot [n-2,\ n-1,\ n] \\ &= [n+4-8i,\ n+5-8i] \odot \alpha_i \odot [1,\ 2,\ 3] \odot \beta_{i+1} \odot \\ &\quad [n-2-8i,\ n-1-8i] \odot [n-4-8i,\ n-3-8i,\ n+2-8i,\ n+3-8i] \odot \\ &\quad [n-8i,\ n+1-8i] \odot \gamma_i \odot [n-2,\ n-1,\ n].\end{aligned}$$

Perceba que a transposição $\tau(1, n-4-4i, n-4i)$ remove o bloco $[n-4-8i,\ n-3-8i,\ n+2-8i,\ n+3-8i]$ de π_i e insere-o de volta no início da permutação, já que o comprimento do sufixo $[n-8i,\ n+1-8i] \odot \gamma_i \odot [n-2,\ n-1,\ n]$ é exatamente $2+4(i-1)+3 = 4i+1$. Temos então

$$\begin{aligned}\pi_{i+1} &= \pi_i \tau(1, n-4-4i, n-4i) \\ &= [n-4-8i,\ n-3-8i,\ n+2-8i,\ n+3-8i] \odot [n+4-8i,\ n+5-8i] \odot \\ &\quad \alpha_i \odot [1,\ 2,\ 3] \odot \beta_{i+1} \odot \\ &\quad [n-2-8i,\ n-1-8i] \odot [n-8i,\ n+1-8i] \odot \gamma_i \odot [n-2,\ n-1,\ n] \\ &= [n-4-8i,\ n-3-8i] \odot [n+2-8i,\ n+3-8i,\ n+4-8i,\ n+5-8i] \odot \\ &\quad \alpha_i \odot [1,\ 2,\ 3] \odot \beta_{i+1} \odot \\ &\quad [n-2-8i,\ n-1-8i,\ n-8i,\ n+1-8i] \odot \gamma_i \odot [n-2,\ n-1,\ n] \\ &= [n-4-8i,\ n-3-8i] \odot \alpha_{i+1} \odot [1,\ 2,\ 3] \odot \beta_{i+1} \odot \gamma_{i+1} \odot [n-2,\ n-1,\ n],\end{aligned}$$

o que prova que após $\lfloor n/8 \rfloor$ transposições da *Fase 3*, para $n \geqslant 8$, nós obtemos a permutação

$$\begin{aligned}\pi_{\lfloor n/8 \rfloor} &= [4+n \bmod 8,\ 5+n \bmod 8] \odot \alpha_{\lfloor n/8 \rfloor} \odot \\ &\quad [1,\ 2,\ 3] \odot \beta_{\lfloor n/8 \rfloor} \odot \gamma_{\lfloor n/8 \rfloor} \odot [n-2,\ n-1,\ n],\end{aligned}$$

onde

$$\begin{aligned}\alpha_{\lfloor n/8 \rfloor} &= \bigodot_{j=0}^{\lfloor n/8 \rfloor - 2} \bigodot_{k=0}^{3} [10+n \bmod 8+8j+k] \\ \beta_{\lfloor n/8 \rfloor} &= \bigodot_{j=0}^{(n \bmod 8)/4-1} [6+4j,\ 7+4j,\ 4+4j,\ 5+4j] \\ \gamma_{\lfloor n/8 \rfloor} &= \bigodot_{j=0}^{\lfloor n/8 \rfloor - 2} \bigodot_{k=0}^{3} [6+n \bmod 8+8j+k].\end{aligned}$$

A transposição $\lfloor n/8 \rfloor + 1$ depende do valor de $n \bmod 8$. Temos dois casos possíveis.

Caso 1: $n \bmod 8 = 0$
Neste caso temos

$$\pi_{\lfloor n/8 \rfloor} = [4,\ 5] \odot \alpha_{\lfloor n/8 \rfloor} \odot [1,\ 2,\ 3] \odot \beta_{\lfloor n/8 \rfloor} \odot \gamma_{\lfloor n/8 \rfloor} \odot [n-2,\ n-1,\ n],$$

onde

$$\begin{aligned}
\alpha_{\lfloor n/8 \rfloor} &= \bigodot_{j=0}^{n/8-2} \bigodot_{k=0}^{3} [10 + 8j + k] \\
\beta_{\lfloor n/8 \rfloor} &= \bigodot_{j=0}^{-1} [6 + 4j,\ 7 + 4j,\ 4 + 4j,\ 5 + 4j] = [\,] \\
\gamma_{\lfloor n/8 \rfloor} &= \bigodot_{j=0}^{n/8-2} \bigodot_{k=0}^{3} [6 + 8j + k].
\end{aligned}$$

Perceba que a transposição $\tau(1, 3, n/2 + 2)$ remove o bloco $[4,\ 5]$ de $\pi_{\lfloor n/8 \rfloor}$ e insere-o de volta logo após o bloco $[1,\ 2,\ 3]$, já que o tamanho do prefixo $[4,\ 5] \odot \alpha_{\lfloor n/8 \rfloor} \odot [1,\ 2,\ 3]$ é exatamente $2 + 4(n/8 - 1) + 3 = n/2 + 1$. Temos então

$$\begin{aligned}
\pi_{\lfloor n/8 \rfloor + 1} &= \pi_{\lfloor n/8 \rfloor} \tau(1, 3, n/2 + 2) \\
&= \alpha_{\lfloor n/8 \rfloor} \odot [1,\ 2,\ 3,\ 4,\ 5] \odot \gamma_{\lfloor n/8 \rfloor} \odot [n-2,\ n-1,\ n] \\
&= \bigodot_{j=0}^{\lceil n/8 \rceil - 2} \bigodot_{k=0}^{3} [10 + 8j + k] \odot [1,\ 2,\ 3,\ 4,\ 5] \odot \\
&\quad \bigodot_{j=0}^{\lfloor n/8 \rfloor - 2} \bigodot_{k=0}^{3} [6 + 8j + k] \odot [n-2,\ n-1,\ n] \\
&= F_3(n).
\end{aligned}$$

Caso 2: $n \bmod 8 = 4$
Neste caso temos

$$\pi_{\lfloor n/8 \rfloor} = [8,\ 9] \odot \alpha_{\lfloor n/8 \rfloor} \odot [1,\ 2,\ 3] \odot \beta_{\lfloor n/8 \rfloor} \odot \gamma_{\lfloor n/8 \rfloor} \odot [n-2,\ n-1,\ n],$$

onde

$$\begin{aligned}
\alpha_{\lfloor n/8 \rfloor} &= \bigodot_{j=0}^{(n-4)/8-2} \bigodot_{k=0}^{3} [14 + 8j + k] \\
\beta_{\lfloor n/8 \rfloor} &= \bigodot_{j=0}^{0} [6 + 4j,\ 7 + 4j,\ 4 + 4j,\ 5 + 4j] = [6,\ 7,\ 4,\ 5] \\
\gamma_{\lfloor n/8 \rfloor} &= \bigodot_{j=0}^{(n-4)/8-2} \bigodot_{k=0}^{3} [10 + 8j + k].
\end{aligned}$$

Perceba que a transposição $\tau(1, n/2, n/2+2)$ remove o bloco $[6,\ 7]$ de $\pi_{\lfloor n/8 \rfloor}$ e insere-o de volta no início da permutação, já que o comprimento do prefixo $[8,\ 9] \odot \alpha_{\lfloor n/8 \rfloor} \odot [1,\ 2,\ 3]$ é exatamente $2 + 4((n-4)/8 - 1) + 3 = n/2 - 1$. Temos então

$$\begin{aligned}
\pi_{\lfloor n/8 \rfloor + 1} &= \pi_{\lfloor n/8 \rfloor} \tau(1, n/2, n/2+2) \\
&= [6,\ 7,\ 8,\ 9] \odot \alpha_{\lfloor n/8 \rfloor} \odot [1,\ 2,\ 3,\ 4,\ 5] \odot \gamma_{\lfloor n/8 \rfloor} \odot [n-2,\ n-1,\ n] \\
&= [6,\ 7,\ 8,\ 9] \odot \bigodot_{j=0}^{(n-4)/8-2} \bigodot_{k=0}^{3} [14 + 8j + k] \odot [1,\ 2,\ 3,\ 4,\ 5] \odot \\
&\quad \bigodot_{j=0}^{(n-4)/8-2} \bigodot_{k=0}^{3} [10 + 8j + k] \odot [n-2,\ n-1,\ n] \\
&= \bigodot_{j=0}^{(n-4)/8-1} \bigodot_{k=0}^{3} [6 + 8j + k] \odot [1,\ 2,\ 3,\ 4,\ 5] \odot \\
&\quad \bigodot_{j=0}^{(n-4)/8-2} \bigodot_{k=0}^{3} [10 + 8j + k] \odot [n-2,\ n-1,\ n] \\
&= \bigodot_{j=0}^{\lceil n/8 \rceil - 2} \bigodot_{k=0}^{3} [6 + 8j + k] \odot [1,\ 2,\ 3,\ 4,\ 5] \odot \\
&\quad \bigodot_{j=0}^{\lfloor n/8 \rfloor - 2} \bigodot_{k=0}^{3} [10 + 8j + k] \odot [n-2,\ n-1,\ n] \\
&= F_3(n),
\end{aligned}$$

o que conclui nossa prova. ■

Lema 3.4 *A* Fase 4 *transforma* $F_3(n)$, *para* $n \geqslant 8$, *na permutação identidade* ι.

Prova: Estamos agora considerando apenas $n \geqslant 8$. Considere π_i, onde $0 \leqslant i \leqslant (n + n \bmod 8)/8 - 1$, como sendo a permutação obtida após i transposições da *Fase 4*. Então temos a seguinte recorrência:

$$\pi_{i+1} = \pi_i \tau(1, 5, 4\lfloor n/8 \rfloor + 6 + 4i).$$

Temos, para a permutação π_i, a seguinte fórmula:

$$\pi_i = \bigodot_{j=0}^{\lceil n/8 \rceil - 2 - i} \bigodot_{k=0}^{3} [10 - n \bmod 8 + 8i + 8j + k] \odot \bigodot_{j=0}^{8 - n \bmod 8 + 8i} [1 + j] \odot \bigodot_{j=0}^{\lceil n/8 \rceil - 2 - i} \bigodot_{k=0}^{3} [14 - n \bmod 8 + 8i + 8j + k].$$

Note que anexamos o elemento $n + 1$ ao final da permutação para simplificar sua representação, o que não interfere com o resultado, já que nenhuma das transposições envolvem esse elemento.

A prova da fórmula acima pode ser feita por indução. Para o caso base $i = 0$, temos, para $n \bmod 8 = 0$,

$$\begin{aligned} \pi_0 &= F_3(n) \\ &= \bigodot_{j=0}^{n/8-2} \bigodot_{k=0}^{3} [10 + 8j + k] \odot [1,\ 2,\ 3,\ 4,\ 5] \odot \\ &\quad \bigodot_{j=0}^{n/8-2} \bigodot_{k=0}^{3} [6 + 8j + k] \odot [n - 2,\ n - 1,\ n]. \end{aligned}$$

Inserindo o elemento $n + 1$ no final:

$$\begin{aligned} \pi_0 &= \bigodot_{j=0}^{n/8-2} \bigodot_{k=0}^{3} [10 + 8j + k] \odot [1,\ 2,\ 3,\ 4,\ 5] \odot \\ &\quad \bigodot_{j=0}^{n/8-2} \bigodot_{k=0}^{3} [6 + 8j + k] \odot [n - 2,\ n - 1,\ n,\ n + 1] \\ &= \bigodot_{j=0}^{\lceil n/8 \rceil - 2} \bigodot_{k=0}^{3} [10 - n \bmod 8 + 8j + k] \odot \bigodot_{j=0}^{8 - n \bmod 8} [1 + j] \odot \\ &\quad \bigodot_{j=0}^{\lceil n/8 \rceil - 2} \bigodot_{k=0}^{3} [14 - n \bmod 8 + 8j + k]. \end{aligned}$$

Para $n \bmod 8 = 4$ temos

$$\begin{aligned}\pi_0 &= F_3(n)\\ &= \bigodot_{j=0}^{(n+4)/8-2} \bigodot_{k=0}^{3} [6+8j+k] \odot [1,\ 2,\ 3,\ 4,\ 5] \odot \\ &\quad \bigodot_{j=0}^{(n-4)/8-2} \bigodot_{k=0}^{3} [10+8j+k] \odot [n-2,\ n-1,\ n].\end{aligned}$$

Inserindo o elemento $n+1$ no final:

$$\begin{aligned}\pi_0 &= \bigodot_{j=0}^{(n+4)/8-2} \bigodot_{k=0}^{3} [6+8j+k] \odot [1,\ 2,\ 3,\ 4,\ 5] \odot \\ &\quad \bigodot_{j=0}^{(n-4)/8-2} \bigodot_{k=0}^{3} [10+8j+k] \odot [n-2,\ n-1,\ n,\ n+1] \\ &= \bigodot_{j=0}^{\lceil n/8\rceil-2} \bigodot_{k=0}^{3} [10-n \bmod 8+8j+k] \odot \bigodot_{j=0}^{8-n \bmod 8} [1+j] \odot \\ &\quad \bigodot_{j=0}^{\lceil n/8\rceil-2} \bigodot_{k=0}^{3} [14-n \bmod 8+8j+k].\end{aligned}$$

Vamos provar agora que, para $1 \leqslant i+1 \leqslant (n+n \bmod 8)/8-1$, temos $\pi_i\tau(1,5,4\lfloor n/8\rfloor + 6+4i) = \pi_{i+1}$. Perceba que a transposição remove o prefixo $\bigodot_{k=0}^{3}[10-n \bmod 8+8i+k]$ e insere-o de volta logo após o bloco $\bigodot_{j=0}^{8-n \bmod 8+8i}[1+j]$, já que o comprimento do prefixo $\bigodot_{j=0}^{\lceil n/8\rceil-2-i} \bigodot_{k=0}^{3}[10-n \bmod 8+8i+8j+k] \odot \bigodot_{j=0}^{8-n \bmod 8+8i}[1+j]$ é exatamente $4((n+n \bmod 8)/8-1-i)+9-n \bmod 8+8i = 4\lfloor n/8\rfloor+5+4i$.

Temos então:

$$\begin{aligned}
\pi_{i+1} &= \pi_i\tau(1,5,4\lfloor n/8\rfloor+6+4i)\\
&= \bigodot_{j=0}^{\lceil n/8\rceil-3-i}\bigodot_{k=0}^{3}[18-n \bmod 8+8i+8j+k]\odot\bigodot_{j=0}^{(8-n \bmod 8+8i)}[1+j]\odot\\
&\quad \bigodot_{k=0}^{3}[10-n \bmod 8+8i+k]\odot\bigodot_{j=0}^{\lceil n/8\rceil-2-i}\bigodot_{k=0}^{3}[14-n \bmod 8+8i+8j+k]\\
&= \bigodot_{j=0}^{\lceil n/8\rceil-3-i}\bigodot_{k=0}^{3}[18-n \bmod 8+8i+8j+k]\odot\bigodot_{j=0}^{16-n \bmod 8+8i}[1+j]\odot\\
&\quad \bigodot_{j=0}^{\lceil n/8\rceil-3-i}\bigodot_{k=0}^{3}[22-n \bmod 8+8i+8j+k]\\
&= \bigodot_{j=0}^{\lceil n/8\rceil-2-(i+1)}\bigodot_{k=0}^{3}[10-n \bmod 8+8(i+1)+8j+k]\odot\bigodot_{j=0}^{8-n \bmod 8+8(i+1)}[1+j]\odot\\
&\quad \bigodot_{j=0}^{\lceil n/8\rceil-2-(i+1)}\bigodot_{k=0}^{3}[14-n \bmod 8+8(i+1)+8j+k],
\end{aligned}$$

o que prova que após as $\lceil n/8-1\rceil$ transposições da *Fase 4* obtemos a permutação

$$\begin{aligned}
\pi_{(n+n \bmod 8)/8-1} &= \bigodot_{j=0}^{-1}[n+2+8j,\ n+3+8j,\ n+4+8j,\ n+5+8j]\odot\bigodot_{j=0}^{n}[1+j]\odot\\
&\quad \bigodot_{j=0}^{-1}[n+6+8j,\ n+7+8j,\ n+8+8j,\ n+9+8j]\\
&= \iota,
\end{aligned}$$

concluindo nossa prova. ■

Com os lemas anteriores na mão, estamos prontos para o seguinte teorema.

Teorema 3.5 *Para $n \geqslant 4$, o Algoritmo 1 ordena a permutação reversa de comprimento n utilizando exatamente $\lceil 3n/4\rceil$ transposições.*

Prova: A prova de que o algoritmo ordena R_n, para $n \geqslant 8$, é resultado direto da aplicação em seqüência dos lemas 3.1, 3.2, 3.3, 3.4. Para $n = 4$ é suficiente executar o algoritmo e verificar que ele realmente ordena R_4.

Agora que sabemos que o Algoritmo 1 de fato ordena R_n, resta saber quantas transposições ele utiliza. Considere $n' = n-(n \bmod 4)$. Como n' é múltiplo 4, $\left\lfloor \frac{n'}{8} \right\rfloor = \frac{n'-n' \bmod 8}{8}$ e $\left\lceil \frac{n'}{8} \right\rceil = \frac{n'+n' \bmod 8}{8}$. Com isso concluímos que o número total de transposições é

$$\begin{aligned} n \bmod 4 + \frac{n'}{4} + \frac{n'}{4} + \left(\left\lfloor \frac{n'}{8} \right\rfloor + 1\right) + \left(\left\lceil \frac{n'}{8} \right\rceil - 1\right) &= \frac{3n'}{4} + n \bmod 4 \\ &= \frac{3n + n \bmod 4}{4} \\ &= \left\lceil \frac{3n}{4} \right\rceil, \end{aligned}$$

pois $x = (3n + n \bmod 4)/4$ é inteiro e satisfaz $\frac{3n}{4} \leqslant x < \frac{3n}{4} + 1$. Isto finaliza nossa prova. ■

Esse teorema nos dá um limite superior para a distância da permutação reversa.

Teorema 3.6 *Para $n \geqslant 1$, temos $d_p(R_n) \leqslant \left\lceil \frac{3n}{4} \right\rceil$*

Prova: Para $n \geqslant 4$, esse limite superior é garantido pelo Teorema 3.5. Para os valores menores de n utilizamos o Teorema 1.11. ■

Christie [9] e Meidanis, Walter e Dias [22] propuseram a seguinte conjectura para o problema geral de ordenação por transposições:

Conjectura 3.7 *O diâmetro de transposição $D(n)$, para $n \geqslant 3$, é dado por*

$$D(n) = d(R_n) = \left\lfloor \frac{n}{2} \right\rfloor + 1.$$

Para $n \leqslant 12$, Eriksson e colegas [12] provaram que essa conjectura é verdadeira. Para $n = 13$ e $n = 15$ eles encontraram contra-exemplos cujas distâncias eram maiores do que a da permutação reversa, e provaram que $D(13) = D(14) = 8$ e $D(15) = 9$. No entanto eles ainda acreditam que a conjectura é verdadeira para $n > 15$, já que seus experimentos computacionais sugerem que os padrões que tornam a ordenação especialmente difícil para $n = 13$ e $n = 15$ cessam de existir para valores maiores de n.

Assim como no problema geral de ordenação por transposição, nós também acreditamos que a seguinte afirmação é verdadeira para o problema análogo de ordenação por transposições de prefixo.

Conjectura 3.8 *O diâmetro de transposição de prefixo $D_p(n)$, para $n \geqslant 4$, é dado por*

$$D_p(n) = d_p(R_n) = \left\lceil \frac{3n}{4} \right\rceil.$$

Podemos dividir a Conjectura 3.8 em duas questões.

$d_p(R_n) = \lceil \frac{3n}{4} \rceil$? Já sabemos que $d_p(R_n) \leqslant \lceil \frac{3n}{4} \rceil$ para $n \geqslant 4$ pelo Teorema 3.6. Se conseguíssemos um limite inferior de igual valor provaríamos a igualdade. Resultados computacionais [10] mostram que tal igualdade é de fato verdade para $n \leqslant 16$.

$D_p(n) = d_p(R_n)$? Pela definição temos que $D_p(n) \geqslant d_p(R_n)$. Se soubermos o valor exato de $d_p(R_n)$ conseguiremos um limite inferior para o diâmetro de transposição de prefixo melhor que o do Teorema 1.21. No entanto ainda faltará um limite superior para concluirmos a igualdade. Resultados computacionais [10] mostraram que tal igualdade é de fato verdade para $n \leqslant 11$.

3.3 Permutações Fáceis

Da mesma forma que tentamos resolver o problema da distância de transposição de prefixo tentando identificar as propriedades que tornam uma ordenação difícil, podemos fazer o mesmo tentando identificar as propriedades que tornam uma ordenação fácil.

Para isso definiremos uma família de permutações que chamaremos de *fáceis*, pois elas possuem ordenação única, com transposições bem determinadas e com a distância igualando-se ao limite inferior por pontos de quebra do Teorema 1.8.

Formalmente, definiremos uma permutação como fácil baseando-se em sua distância e número de pontos de quebra. As outras propriedades serão conseqüência dessa definição.

Definição Uma permutação π é considerada *fácil* se, e somente se, $d_p(\pi) = \frac{b_p(\pi)-1}{2}$.

Repare que, como a distância é um valor inteiro, então toda permutação fácil deve ter um número ímpar de pontos de quebra de prefixo.

A definição apresentada permite uma caracterização recursiva equivalente de permutação fácil.

Teorema 3.9 *Uma permutação π é fácil se, e somente se, já está ordenada ou existe uma transposição de prefixo τ tal que $b_p(\pi\tau) = b_p(\pi) - 2$ e $\tau\pi$ é fácil.*

Prova: Se a permutação π já está ordenada, então ela é fácil. Vamos considerar então que π não está ordenada.

Precisamos primeiro provar que, se existe uma transposição de prefixo τ tal que $b_p(\pi\tau) = b_p(\pi) - 2$ e $\tau\pi$ é fácil, então π é fácil. Temos que $d_p(\pi) \leqslant d_p(\pi\tau) + 1 = (b_p(\pi\tau) - 1)/2 + 1 = (b_p(\pi) - 1)/2$. No entanto, pelo Teorema 1.8, vale a igualdade e, portanto, π é fácil.

Vamos agora provar que, se π é fácil, existe uma transposição de prefixo τ tal que $b_p(\pi\tau) = b_p(\pi) - 2$ e $\pi\tau$ é fácil. Considere uma transposição de prefixo τ tal que $d_p(\pi\tau) = d_p(\pi) - 1$. Pelo Lema 1.7 temos $b_p(\pi) \leqslant b_p(\pi\tau) + 2$. Como π é fácil, então $d_p(\pi\tau) =$

$(b_p(\pi) - 1)/2 - 1 \leqslant (b_p(\pi\tau) - 1)/2$. No entanto, pelo Teorema 1.8, vale a igualdade e, portanto, $\pi\tau$ é fácil e $b_p(\pi\tau) = b_p(\pi) - 2$. ■

3.3.1 Propriedades

Como visto no Lema 1.7, não podemos retirar mais do que dois pontos de quebra de prefixo com uma transposição de prefixo. Por isso somos obrigados a retirar exatamente dois pontos de quebra de prefixo em cada passo de uma ordenação ótima de uma permutação fácil. Veremos a seguir que, se existe uma transposição de prefixo que retira dois pontos de quebra de uma permutação, tal transposição é única.

Lema 3.10 *Existe no máximo uma transposição de prefixo $\tau(1, j, k)$ que retira dois pontos de quebra de uma permutação π. Caso exista, então $k = \pi^{-1}(pred(\pi(1))) + 1$ e $j = \pi^{-1}(pred(\pi(k))) + 1$.*

Prova: Pela Proposição 1.6, para termos $b_p(\pi\tau) - b_p(\pi) = -2$, com $\tau = \tau(1, j, k)$, devemos ter $\pi(j-1) \neq pred(\pi(j))$, $\pi(k-1) \neq pred(\pi(k))$, $\pi(k-1) = pred(\pi(1))$ e $\pi(j-1) = pred(\pi(k))$. Com isso, $k = \pi^{-1}(pred(\pi(1))) + 1$ e $j = \pi^{-1}(pred(\pi(k))) + 1$ e tal solução, se possível, é única. ■

E isso nos garante uma propriedade muito especial para as permutações fáceis:

Teorema 3.11 (Dias e Meidanis [10]) *Toda permutação fácil possui uma única forma de ordenação ótima por transposições de prefixo.*

Prova: Como temos que tirar exatamente dois pontos de quebra em todo o passo de uma ordenação ótima de uma permutação especial e pelo Lema 3.10 existe uma única transposição de prefixo em cada passo que consegue isso, então só temos uma seqüência de transposições de prefixo que ordena otimamente uma permutação fácil. ■

Corolário 3.12 *Todas as permutações intermediárias de uma ordenação ótima por transposições de prefixo de uma permutação fácil são fáceis.*

Prova: Se temos uma permutação fácil π e aplicamos uma transposição τ que retira dois pontos de quebra de π, então pelos Lema 3.10 e Teorema 3.11 $d_p(\pi\tau) = d_p(\pi) - 1$. No entanto também temos que $b_p(\tau\pi) = b_p(\pi) - 2$, o que nos dá $d_p(\pi\tau) = (b_p(\pi\tau) - 1)/2$ e, portanto, $\pi\tau$ também é fácil. ■

Uma das conseqüências do Corolário 3.12 é que toda ordenação ótima pode ser dividida em duas etapas. Inicialmente passamos apenas por permutações não fáceis e, em seguida,

passamos apenas por permutações fáceis. Caso a permutação a ser ordenada seja fácil, então não há a primeira etapa, pois só passamos por permutações fáceis em sua ordenação. Já a segunda etapa sempre existe, pois temos que chegar na permutação ordenada, que é fácil.

Teorema 3.13 *É possível ordenar otimamente por transposições de prefixo uma permutação fácil em tempo polinomial.*

Prova: Se queremos ordenar uma permutação fácil π e ela já está ordenada, estão o problema já está resolvido. Caso contrário, pelo Teorema 3.9, existe uma transposição de prefixo τ tal que $b_p(\pi\tau) = b_p(\pi) - 2$ e $\pi\tau$ é fácil. Obtemos essa transposição τ em tempo $O(1)$ como descrito no Lema 3.10. Como $d_p(\pi\tau) = d_p(\pi) - 1$ e $\pi\tau$ é fácil, aplicamos esse procedimento recursivamente, completando a indução e obtendo um tempo total $\Theta(d_p(\pi) \times$ (tempo de uma transposição)), que é facilmente implementado como $O(n^2)$, onde n é o comprimento da permutação π. ■

Dias e Meidanis [10] publicaram anteriormente um teorema equivalente:

Teorema 3.14 *É possível determinar se uma permutação é fácil em tempo polinomial.*

Prova: Se queremos testar uma permutação π e ela estiver ordenada, então π é fácil. Caso contrário, buscamos uma transposição de prefixo τ tal que $b_p(\pi\tau) = b_p(\pi) - 2$ como descrito no Lema 3.10. Pelo Teorema 3.9, se não houver tal transposição, então π não é fácil. Se houver tal transposição, então π é fácil se, e somente se, $\pi\tau$ é fácil. Testamos então $\pi\tau$ recursivamente, completando a indução. Isso nos dá um algoritmo $O(n^2)$ para reconhecimento de permutações fáceis, onde n é o comprimento da permutação testada. ■

Como dito anteriormente, em uma ordenação ótima temos duas etapas: passamos inicialmente apenas por permutações não fáceis e, em seguida, apenas por permutações fáceis. Se chegarmos na segunda etapa, então, pelo Teorema 3.13, o problema da ordenação por transposição de prefixo fica fácil. Com isso, uma forma de se tentar resolver o problema da ordenação de uma permutação por transposição de prefixo é encontrar qual permutação fácil está em sua ordenação ótima e como chegar otimamente nela. Baseando-se nessa idéia propomos a seguinte conjectura.

Conjectura 3.15 *Para toda permutação π de comprimento n existe uma permutação fácil π', tal que $b_p(\pi') = b_p(\pi) - k$ e*

$$d_p(\pi, \pi') \leqslant \left\lceil \frac{n}{4} + \frac{k}{2} \right\rceil .$$

A corretude de tal conjectura implica no seguinte limite superior para o diâmetro de transposição de prefixo.

Teorema 3.16 *Se a Conjectura 3.15 for verdadeira, então* $D_p(n) \leqslant \lceil \frac{3n}{4} \rceil$.

Prova: Seja π uma permutação qualquer de comprimento n. Pela Conjectura 3.15 existe uma permutação π', tal que $b_p(\pi') = b_p(\pi) - k$ e $d_p(\pi, \pi') \leqslant \lceil \frac{n}{4} + \frac{k}{2} \rceil$. Como π' é fácil, $d_p(\pi') = \frac{b(\pi')-1}{2} = \frac{b_p(\pi)-k-1}{2}$. Com isso temos

$$d_p(\pi) \leqslant d_p(\pi, \pi') + d_p(\pi') \leqslant \left\lceil \frac{n}{4} + \frac{k}{2} \right\rceil + \frac{b_p(\pi) - k - 1}{2} \leqslant \left\lceil \frac{3n}{4} \right\rceil$$

para toda permutação π de comprimento n e, portanto, $D_p(n) \leqslant \lceil \frac{3n}{4} \rceil$. ■

Note que esse limite superior para o diâmetro coincide com o nosso limite superior para a distância da permutação reversa (Teorema 3.6), complementando as observações feitas na apresentação da Conjectura 3.8. Cabe notar também que as permutações $F_1(n)$, $F_2(n)$ e $F_3(n)$, com n múltiplo de 4, são todas facilmente ordenáveis e, dessa forma, validam a Conjectura 3.15 para o caso da permutação reversa. Para verificar isso tome $\pi = R_n$ e $\pi' = F_1(n - n \bmod 4)$. Dessa forma $b_p(\pi') = b_p(\pi) - n \bmod 4$ e

$$\begin{aligned} d_p(\pi, \pi') &= n \bmod 4 + \frac{n - n \bmod 4}{4} \\ &= \frac{n + 3(n \bmod 4)}{4} = \left\lfloor \frac{n + 3(n \bmod 4)}{4} \right\rfloor \\ &\leqslant \left\lfloor \frac{n + 2(n \bmod 4) + 3}{4} \right\rfloor = \left\lceil \frac{n + 2(n \bmod 4)}{4} \right\rceil \\ &= \left\lceil \frac{n}{4} + \frac{n \bmod 4}{2} \right\rceil. \end{aligned}$$

Para conseguirmos transformar permutações quaisquer em fáceis com poucos movimentos é interessante conhecer melhor a estrutura de uma permutação fácil. Na busca de melhor caracterizarmos as permutações fáceis obtivemos as seguintes propriedades.

Lema 3.17 *Para toda permutação fácil* π *que não esteja ordenada, temos* $pred(\pi(1)) \neq 0$.

Prova: Como $d_p(\pi) \geqslant 1$, então, pelo Teorema 3.9 e Lema 3.10, existe uma transposição única $\tau(1, j, k)$ tal que $d(\pi\tau) = d(\pi) - 1$ e $b_p(\pi\tau) = b_p(\pi) - 2$. Nesse caso, $k = \pi^{-1}(pred(\pi(1))) + 1$ e $j = \pi^{-1}(pred(\pi(k)), \pi) + 1$. Se $pred(\pi(1)) = 0$, então $k = \pi^{-1}(0) + 1 = 1$, o que é absurdo, pois $k > j > 1$. Então necessariamente $pred(\pi(1)) \neq 0$. ■

Note que esse lema implica em $b(\pi) = b_p(\pi)$ para toda permutação fácil π não ordenada.

Teorema 3.18 *Toda permutação fácil possui no máximo um ciclo não trivial em seu diagrama de ciclos.*

Prova: Considere uma permutação π. Podemos provar esse teorema por indução na distância de transposição de prefixo $d_p(\pi)$.

Se $d_p(\pi) = 0$, então π já está ordenado e portanto só possui ciclos triviais.

Se $d_p(\pi) \geqslant 1$, então, pelo Teorema 3.9, existe uma transposição de prefixo τ tal que $b_p(\pi\tau) = b_p(\pi) - 2$ e $\pi\tau$ é fácil. Por hipótese de indução, $c(\pi\tau) - c_t(\pi\tau) \leqslant 1$. Pelo Lema 3.17 e Proposição 1.13, $c_t(\pi\tau) - c_t(\pi) = 2$. No entanto só é possível criar dois novos ciclos se os três pontos onde a transposição τ quebra a permutação π estão em um mesmo ciclo de π. Nesse caso devemos ter obrigatoriamente $c(\pi) = c(\pi\tau) - 2$. Com isso temos $c(\pi) - c_t(\pi) = (c(\pi\tau) - 2) - (c_t(\pi\tau) - 2) = c(\pi\tau) - c_t(\pi\tau) \leqslant 1$, concluindo nossa prova. ■

Embora não tenhamos conseguido encontrar meios de transformar eficientemente permutações quaisquer em fáceis por meio dessas propriedades, encontramos uma ligação entre os problemas de distância de transposição e distância de transposição de prefixo. Com tais propriedades podemos provar que uma ordenação ótima de uma permutação fácil no problema de transposições de prefixo também é uma ordenação ótima no problema de transposições original.

Teorema 3.19 *Para toda permutação fácil π, temos $d_p(\pi) = d(\pi)$.*

Prova: Se $d_p(\pi) = 0$ o caso é trivial. Assumiremos então que $d_p(\pi) \geqslant 1$.

Pelo Teorema 1.15 temos

$$\frac{n+1-c(\pi)}{2} \leqslant \frac{n+1-c_{odd}(\pi)}{2} \leqslant d(\pi).$$

Como $d_p(\pi) \geqslant 1$, pelo Lema 3.17

$$d_p(\pi) = \frac{b_p(\pi)-1}{2} = \frac{b(\pi) + [[pred(\pi(1)) = 0]] - 1}{2} = \frac{b(\pi)-1}{2}.$$

Pela Proposição 1.13 e Teorema 3.18

$$d_p(\pi) = \frac{n - c_t(\pi)}{2} \leqslant \frac{n+1-c(\pi)}{2}.$$

Com isso temos

$$\frac{n+1-c(\pi)}{2} \leqslant d(\pi) \leqslant d_p(\pi) \leqslant \frac{n+1-c(\pi)}{2}$$

e, portanto, $d(\pi) = d_p(\pi)$. ■

Esse teorema é um dos principais resultados desta dissertação, pois é um ponto de união entre os problemas de distância de transposição e distância de transposição de prefixo, definindo uma nova família de permutações que podem ser ordenadas eficientemente em ambos os problemas. Essa é uma das maiores famílias de permutações que se sabe calcular eficientemente suas distâncias de transposição. Tal resultado vai ao encontro da proposta de Christie [8] de tentar resolver o problema da distância de transposição por meio da construção de um conjunto de famílias de permutações cujas distâncias são bem conhecidas e que cubra todas as possíveis permutações.

Com isso fica claro que o problema da distância de transposição de prefixo pode realmente ajudar no estudo do problema da distância de transposição, apesar de não estar claro se um pode ser resolvido pelo outro por redução polinomial.

3.3.2 Contagem

Uma possibilidade que foi considerada para tentar resolver o problema da distância por transposições de prefixo foi a de que poderia haver um número polinomial de permutações que não são fáceis. Se isso fosse verdade, poderíamos calcular a distância de uma permutação por meio de uma busca em largura que parasse sempre que encontrássemos permutações fáceis. Nesse caso percorreríamos, no pior caso, um número polinomial de permutações, o que nos daria um algoritmo também polinomial.

Isso motivou a busca pelo número de permutações fáceis de comprimento n que, como veremos, cresce muito rapidamente com o aumento de n, mas, infelizmente, a diferença entre $n!$ e esse valor cresce ainda mais rapidamente.

O que nos ajuda na contagem das permutações fáceis é o fato de sempre existir no máximo uma transposição de prefixo que remove dois pontos de quebra, como expresso pelo Lema 3.10.

Vamos realizar a contagem começando pela permutação com distância zero, seguida pelas permutações com distância um, e assim por diante. O princípio básico por trás disso é o seguinte. Se temos uma permutação fácil π e aplicamos uma transposição de prefixo τ que aumenta em dois o número de pontos de quebra de prefixo, então $\pi\tau$ também é fácil. Isso vem do Teorema 3.9, pois existe a transposição de prefixo τ^{-1} que retira dois pontos de quebra de prefixo de $\pi\tau$, obtendo a permutação fácil π.

Agora vem a pergunta: *Quantas permutações fáceis eu consigo obter adicionando dois pontos de quebra de prefixo a uma permutação fácil por meio de uma transposição de prefixo?* Basta contar quantas transposições de prefixo existem que adicionam dois pontos de quebra de prefixo à permutação, pois cada transposição gera uma permutação diferente. Pela Proposição 1.6, a aplicação de uma transposição de prefixo $\tau(1, j, k)$ a uma permutação π aumenta em dois os pontos quebra de prefixo se, e somente, $pred(\pi(j), \pi) =$

$pred(\pi(j))$ e $pred(\pi(k), \pi) = pred(\pi(k))$. Isso significa que todos os possíveis candidatos a j e k são justamente aqueles valores i tais que $\pi(i)$ não é um ponto de quebra de prefixo. Com isso, para uma permutação fácil π de comprimento n temos exatamente $n+1-b_p(\pi)$ candidatos, ou seja $\binom{n+1-b_p(\pi)}{2}$ transposições que aumentam em dois o número de pontos de quebra de prefixo. Note que esse valor depende apenas do comprimento e do número de pontos de quebra de prefixo da permutação, sendo que este último pode ser substituído por $2d+1$, onde d é a distância de prefixo.

Uma outra observação importante é que, a partir de duas permutações fáceis diferentes, adicionar dois pontos de quebra de prefixo por meio de uma transposição de prefixo a cada uma delas nunca resulta em duas permutações iguais, pois, caso contrário, isso violaria a ordenação única (Teorema 3.11).

Com isso podemos obter facilmente o número de permutações fáceis com distância de transposição de prefixo d a partir do número permutações fáceis com distância de transposição de prefixo $d-1$. Seja $\mathcal{F}(n,d)$ o número de permutações fáceis de comprimento n e distância de transposição de prefixo d. Sabemos que tais permutações possuem $b = 2d+1$ pontos de quebra de prefixo. Então

$$\mathcal{F}(n,d+1) = \binom{n+1-b}{2}\mathcal{F}(n,d) = \binom{n-2d}{2}\mathcal{F}(n,d) = \frac{(n-2d)(n-2d-1)}{2}\mathcal{F}(n,d),$$

onde $1 \leqslant d+1 \leqslant \frac{n}{2}$ e sendo que $\mathcal{F}(n,0) = 1$.

Desenvolvendo a recorrência obtemos

$$\begin{aligned}
\mathcal{F}(n,d) &= \frac{(n-2d+2)(n-2d+1)}{2}\mathcal{F}(n,d-1) \\
&= \prod_{i=1}^{d}\frac{(n-2i+2)(n-2i+1)}{2} \\
&= \frac{1}{2^d}\prod_{i=n-2d+1}^{n} i \\
&= \frac{n!}{2^d(n-2d)!}.
\end{aligned}$$

Portanto, para calcular o número $\mathcal{F}(n)$ de todas as permutações fáceis de comprimento n temos

$$\mathcal{F}(n) = \sum_{d=0}^{\lfloor n/2 \rfloor}\frac{n!}{2^d(n-2d)!}.$$

Podemos ver alguns dos possíveis valores de $\mathcal{F}(n,d)$ na Tabela 3.1.

Comparando $\mathcal{F}(n)$ com $n!$, veremos que a razão entre ambos diminui rapidamente com o crescimento de n, conforme ilustrado pelo gráfico da Figura 3.2. De $n = 5$ em diante,

n	$d=0$	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$	$d=6$	$\mathcal{F}(n)$
1	1	0	0	0	0	0	0	1
2	1	1	0	0	0	0	0	2
3	1	3	0	0	0	0	0	4
4	1	6	6	0	0	0	0	13
5	1	10	30	0	0	0	0	41
6	1	15	90	90	0	0	0	196
7	1	21	210	630	0	0	0	862
8	1	28	420	2520	2520	0	0	5489
9	1	36	756	7560	22680	0	0	31033
10	1	45	1260	18900	113400	113400	0	247006
11	1	55	1980	41580	415800	1247400	0	1706816
12	1	66	2970	83160	1247400	7484400	7484400	16302397

Tabela 3.1: Número de permutações fáceis para diferentes comprimentos e distâncias

menos da metade das permutações de comprimento n são fáceis. Com isso podemos concluir que $\frac{n!}{2} \leqslant n! - \mathcal{F}(n) \leqslant n!$ para valores grandes de n e, portanto, o número de permutações que não são fáceis é $\Theta(n!)$.

3.4 Conclusões e Problemas em Aberto

Neste capítulo vimos inicialmente como ordenar a permutação reversa. O algoritmo apresentado utiliza $\lceil 3n/4 \rceil$ transposições de prefixo para ordenar R_n. Para n múltiplo de 4, o que ele faz é transformar a permutação R_n em uma permutação fácil com o mesmo número de pontos de quebra utilizando $n/4$ passos. Não foi provado que tal ordenação é ótima, mas o resultado encontrado nos dá um limite superior para a distância de transposição de prefixo da permutação reversa, que conjectura-se que seja igual ao diâmetro de transposição de prefixo.

A seguir vimos as permutações fáceis de ordenar, que são aquelas que alcançam o limite inferior por pontos de quebra para a distância de transposição de prefixo (Teorema 1.8). Encontramos diversas propriedades interessantes das permutações fáceis: possuem ordenação ótima única, podem ser ordenadas otimamente em tempo polinomial, possuem apenas um ciclo não trivial. Sugerimos uma forma de resolver o problema da distância de transposição de prefixo em que temos que descobrir como transformar eficientemente uma permutação qualquer em uma fácil. Infelizmente as propriedades encontradas não foram suficientes para tornar claro como fazer isso, mas enunciamos uma conjectura que estabelece uma propriedade que tal transformação deve ter para obtermos um limite superior

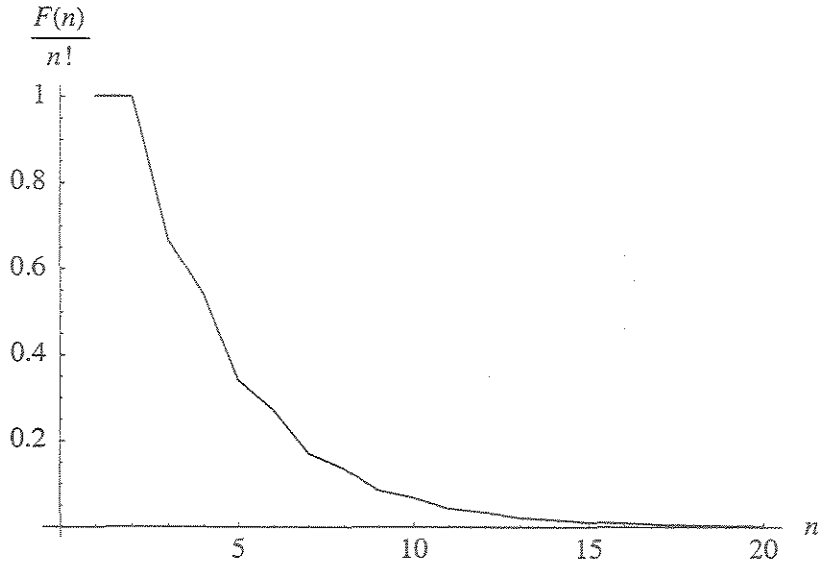

Figura 3.2: Razão entre $\mathcal{F}(n)$ e $n!$.

mais justo para o diâmetro de transposição de prefixo.

Como forma de encontrar mais propriedades que permitam uma caracterização mais conveniente das permutações fáceis, podemos conjecturar a seguinte definição alternativa.

Conjectura 3.20 *Uma permutação é fácil se, e somente se, possui uma única forma de ordenação ótima.*

Tal fato pôde ser verificado positivamente por meios computacionais para todas as permutações de comprimento menor ou igual a 5.

Uma outra propriedade que pode ajudar na caracterização das permutações fáceis é o fato de ela sempre admitir uma transposição que retira dois pontos de quebra. Permutações que admitem tais transposições possuem pelo menos um ciclo chamado de *totalmente orientado* por Christie, que, em sua tese de doutorado [9], estabelece diversas propriedades sobre eles. No entanto a permutação possuir um ciclo totalmente orientado é uma condição necessária, mas não suficiente para ser fácil.

Um fato interessante que foi descoberto é que a distância de transposição se iguala à distância de transposição de prefixo para as permutações fáceis, servindo como um ponto de ligação entre os dois problemas. Podemos tentar generalizar o conceito de permutações fáceis para o problema da distância de transposição. Uma possibilidade é considerar como fáceis as permutações que alcançam o limite inferior de Bafna e Pevzner (Teorema 1.18), no entanto é fácil ver que perde-se a propriedade de ordenação única. Uma outra possibilidade é classificar como fáceis os ciclos, ao invés das permutações. No caso de transposição de prefixo, temos justamente um ciclo que conseguimos sempre

quebrar em três ciclos criando dois ciclos triviais. No caso de transposição poderíamos ter vários ciclos com essa propriedade. Uma dificuldade é que pode ser que a ordenação de um ciclo afete a ordenação dos outros.

Para finalizar o assunto das permutações fáceis realizamos a contagem delas. Infelizmente observamos que existem muito mais permutações não fáceis do que fáceis.

Capítulo 4

Conclusão

Neste trabalho abordamos duas medidas de distância entre permutações: a distância de transposição e a distância de transposição de prefixo. O problema que considera somente transposições de prefixo surgiu como forma de estudar a distância de transposição sob outro ponto de vista. Tal abordagem foi provada útil.

Muito ainda há para se desvendar sobre o problema da distância de transposição, pois ele continua em aberto. No sentido de resolvê-lo, demos algumas contribuições originais.

Para o problema da distância de transposição tivemos as seguintes principais contribuições.

- Introduzimos a ferramenta de remoção de elementos, utilizamos análise de subseqüências e sugerimos novas formas de se explorar esses conceitos.
- Encontramos uma família não trivial de permutações que podem ser ordenadas em tempo polinomial (permutações fáceis).

Como trabalhos futuros, podemos tentar aproveitar as sugestões dadas para explorar o uso da remoção de elementos e análise nas subseqüências. É interessante notar que utilizando esses conceitos muitas vezes conseguimos obter resultados melhores considerando permutações circulares ou tóricas, como explicado por Eriksson e colegas [12].

Podemos também tentar ampliar o conceito de permutações fáceis e assim estender a família de permutações para as quais se conhece algoritmo polinomial para ordenação por transposições.

Para o problema da distância de transposição de prefixo, tivemos as seguintes principais contribuições.

- Elaboramos um algoritmo polinomial que ordena a permutação reversa de comprimento n com $\lceil 3n/4 \rceil$ transposições de prefixo.
- Enumeramos algumas propriedades das permutações fáceis.

- Calculamos o número de permutações fáceis.

Como trabalhos futuros, podemos tentar resolver as conjecturas propostas sobre a distância de transposição de prefixo da permutação reversa e o diâmetro de transposição de prefixo.

Outra possibilidade é tentar caracterizar melhor as permutações fáceis, encontrando novas propriedades que tornem possível encontrar o número de transposições de prefixo necessárias para se chegar a uma permutação fácil a partir de uma qualquer de forma eficiente. Isso poderia nos garantir um algoritmo de aproximação com fator bastante justo.

Apêndice A

Implementações

A.1 *Mathematica*

Durante o desenvolvimento deste projeto de mestrado havia a necessidade constante de se manipular permutações e calcular parâmetros sobre elas. Esses procedimentos eram realizados manualmente e se perdia muito tempo com isso. Procuramos, então, uma ferramenta que permitisse automatizar tais operações. A opção que se mostrou mais adequada foi o software *Mathematica*, da companhia *Wolfram Research*.

O software *Mathematica* é uma poderosa ferramenta matemática que é utilizada em diversas áreas do conhecimento tais como Química, Engenharia Química, Ciência da Computação, Engenharia de Controle, Economia, Estatística, Engenharia Elétrica, Medicina, Matemática, Física, Engenharia Civil, Engenharia Mecânica, entre outras.

Essa grande abrangência de uso é devida às diversas funcionalidades que o software possui, além de sua grande flexibilidade. Entre seus usos e funcionalidades podemos citar:

- Calculadora avançada.
- Manipulação simbólica de cálculos complexos.
- Leitura, análise e visualização de dados.
- Resolução de equações, equações diferenciais e problemas de minimização tanto simbólica como numericamente.
- Ambiente de modelagem matemática e simulação de diversos tipos de sistemas.
- Linguagem de programação.
- Criação rápida de novos softwares científicos por meio da reutilização de suas bibliotecas.

Diante de tantos recursos tal software se mostrou muito adequado para nos auxiliar na manipulação de permutações, análise de casos, teste de novas idéias e conjecturas e aperfeiçoamento de nossa intuição, mesmo sendo necessária a aquisição de uma licença de uso.

O *Mathematica* possui um bom suporte para manipulação de permutações e visualização de dados. No entanto não possui funções específicas para o uso no contexto de rearranjo de genomas. Para contornar essa situação elaboramos uma biblioteca com diversas funcionalidades para se utilizar nessa área.

Representamos permutações por expressões cujo *head* é `Permutation`. Assim podemos declarar permutações das seguintes formas.

```
p = Permutation[2, 3, 4, 1, 5];
q = Permutation @@ {1, 3, 4, 2, 5};
```

Para obter um elemento, basta utilizar o operador `Part` (`[[ ]]`).

```
In[5]:= p[[2]]
Out[5]= 3
```

Para utilizar transposições, definimos a expressão `Tau[i,j,k]` e redefinimos o operador de multiplicação não comutativa (`**`).

```
In[7]:= p ** Tau[1,3,4]
        % ** Tau[2,4,5]
Out[7]= Permutation[4,2,3,1,5]
Out[8]= Permutation[4,1,2,3,5]
```

Note que não precisamos passar o comprimento `Length[p]` da permutação `p` para a transposição `Tau`.

Podemos facilmente encontrar o número de pontos de quebra (`tb`) e o número de pontos de quebra de prefixo(`ptb`) entre duas permutações dadas ou entre uma permutação e ela ordenada.

```
In[9]:= tb[p, p**Tau[2,3,4]]
Out[9]= 3

In[10]:= ptb[q, Permutation@@Range[5]]
Out[10]= 4

In[11]:= ptb[q]
Out[11]= 4
```

A notação adotada foi inspirada na de Christie [9]. Utilizamos o prefixo t para indicar que a função se refere a transposição e pt para transposição de prefixo. Poderíamos a princípio deixar o t de lado, mas a idéia é de que a biblioteca possa ser estendida para outros rearranjos, como reversão por exemplo.

Para testar se duas permutações são vizinhas (distância um), utilizamos `NeighbourQ`.

```
In[12]:= With[{u=Permutation[2,3,5,4,1], v=Permutation[2,4,3,5,1]},
           {tNeighbourQ[u,v], ptNeighbourQ[u,v]}
         ]
Out[12]= {True,False}
```

Para testar se uma permutação é fácil utilizamos `EasyQ`.

```
In[12]:= EasyQ[Permutation[2, 8, 7, 4, 3, 6, 5, 1]]
Out[12]= True
In[13]:= EasyQ[Permutation[8, 7, 6, 5, 4, 3, 2, 1]]
Out[13]= False
```

Utilizando essa função foi possível conferir a fórmula de contagem das permutações especiais para permutações de tamanho pequeno.

Podemos imprimir as permutações de forma mais conveniente com `PrintP`. O diagrama apresentado exibe os pontos de quebra, como ilustra a Figura A.1.

```
PrintP[ Permutation@@Reverse[Range[16]] ** Tau[1, 2, 3] ]
```

	15	16	14	13	12	11	10	9	8	7	6	5	4	3	2	1	

Figura A.1: Uso da função `PrintP`.

Essa forma de visualizar as permutações foi muito útil principalmente na elaboração do Algoritmo 1 de ordenação da permutação reversa, pois precisávamos caracterizar as transposições e as permutações intermediárias em cada passo. Para isso implementamos em *Mathematica* uma variação do algoritmo em que tínhamos explícitas apenas as transposições das fases 0 e 1. A partir da fase 2 calculávamos as transposições como demonstrado no Lema 3.10, pois elas sempre removiam dois pontos de quebra. Imprimindo as permutações e transposições para diversos casos, como ilustra a Figura 3.1, conseguimos caracterizar ambas em todos os passos do algoritmo. Utilizando a implementação de nosso algoritmo em *Mathematica* foi possível provar sua corretude para permutações reversas com vários milhares de elementos.

Uma outra forma de visualizar permutações que é muito importante é o diagrama de ciclos. Elaboramos uma função `CycleDiagram` que gera automaticamente tal diagrama para qualquer permutação. A Figura A.2 exibe diagramas gerados por tal função.

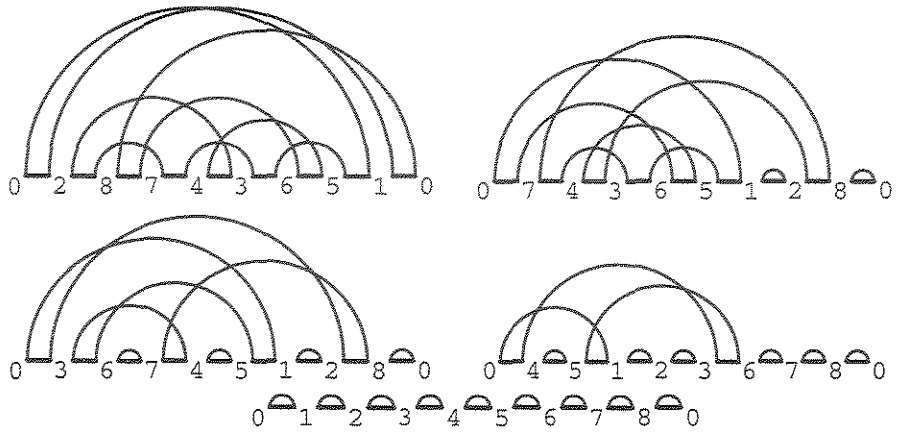

Figura A.2: Diagrama de ciclos das permutações intermediárias na ordenação de $F_1(8)$.

Um recurso do *Mathematica* que se mostrou interessante é a manipulação de grafos. Isso nos permitiu construir e visualizar grafos em que os nós são permutações e as arestas ligam pares de permutações cuja distância é um, podendo essa distância ser de transposição ou transposição de prefixo. A Figura A.3 foi gerada pelo *Mathematica* e ilustra tal grafo para permutações de tamanho 4 utilizando distância de transposição de prefixo. De tal grafo é possível extrair diversas informações úteis, como a distância entre permutações, e testar conjecturas. Foi possível ver, para permutações de comprimento até 7, que a permutação reversa possui sempre a distância igual ao diâmetro e que existem diversas outras permutações que também possuem essa distância. Infelizmente não foi possível gerar o grafo para permutações maiores que 7 devido às restrições de memória e tempo de execução.

A biblioteca em *Mathematica* desenvolvida neste trabalho está disponível na Internet no endereço `www.viniciusf.cjb.net` e é de uso livre para fins acadêmicos.

A.2 Distribuição Distância × Pontos de quebra

Como as restrições das ferramentas em Mathematica que não permitiam elaborar algoritmos que envolvessem todas as permutações de tamanho n, com $n > 7$, decidimos implementar um programa em C++ mais eficiente para buscar mais propriedades sobre o conjunto de permutações.

Em particular, buscamos o número de permutações para cada par (distância, pontos de quebra de prefixo) e a distância média. Curiosamente a distância média aparenta estar sempre em torno de $0.6(n-1)$ para permutações de comprimento n. Os resultados estão nas tabelas a seguir. Eriksson e colegas [12] mostram que a distância de transposição de permutações aleatórias é na média $\left\lceil \frac{n+1}{2} \right\rceil$.

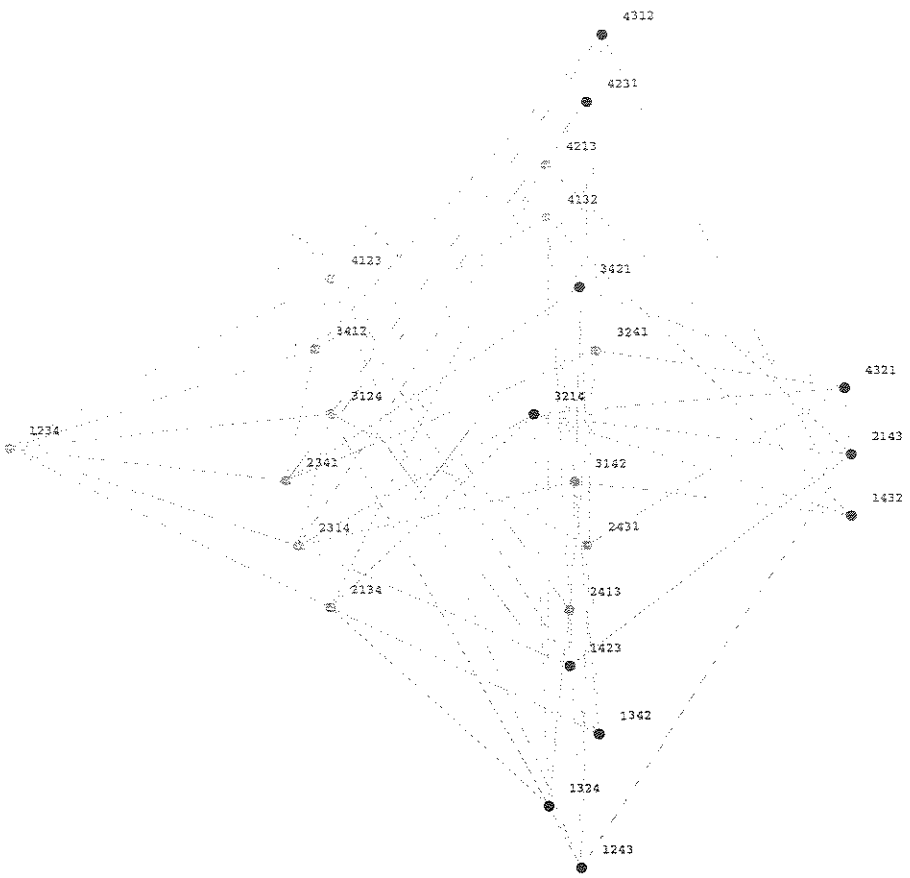

Figura A.3: Grafo de distâncias de transposição de prefixo das permutações de tamanho 4. As permutações fáceis estão indicadas por vértices de cor mais clara.

n	Distância média	Distância média/$(n-1)$
2	0.5	0.5
3	1.16666666666667	0.583333333333333
4	1.79166666666667	0.597222222222222
5	2.425	0.60625
6	3.05694444444444	0.611388888888889
7	3.67579365079365	0.612632275132275
8	4.29097222222222	0.612996031746032
9	4.89711750440917	0.612139688051146
10	5.49496720679012	0.610551911865569
11	6.08778955226872	0.608778955226872

Tabela A.1: Distância média.

n = 2	d = 0	d = 1	Total
b = 1	1	0	1
b = 2	0	0	0
b = 3	0	1	1
Total	1	1	2

Tabela A.2: Distribuição das permutações de comprimento 2. O valor b representa o número de pontos de quebra de prefixo e d a distância de transposição de prefixo.

n = 3	d = 0	d = 1	d = 2	Total
b = 1	1	0	0	1
b = 2	0	0	0	0
b = 3	0	3	0	3
b = 4	0	0	2	2
Total	1	3	2	6

Tabela A.3: Distribuição das permutações de comprimento 3. O valor b representa o número de pontos de quebra de prefixo e d a distância de transposição de prefixo.

n = 4	d = 0	d = 1	d = 2	d = 3	Total
b = 1	1	0	0	0	1
b = 2	0	0	0	0	0
b = 3	0	6	0	0	6
b = 4	0	0	8	0	8
b = 5	0	0	6	3	9
Total	1	6	14	3	24

Tabela A.4: Distribuição das permutações de comprimento 4. O valor b representa o número de pontos de quebra de prefixo e d a distância de transposição de prefixo.

n = 5	d = 0	d = 1	d = 2	d = 3	d = 4	Total
b = 1	1	0	0	0	0	1
b = 2	0	0	0	0	0	0
b = 3	0	10	0	0	0	10
b = 4	0	0	20	0	0	20
b = 5	0	0	30	15	0	45
b = 6	0	0	0	40	4	44
Total	1	10	50	55	4	120

Tabela A.5: Distribuição das permutações de comprimento 5. O valor b representa o número de pontos de quebra de prefixo e d a distância de transposição de prefixo.

n = 6	d = 0	d = 1	d = 2	d = 3	d = 4	d = 5	Total
b = 1	1	0	0	0	0	0	1
b = 2	0	0	0	0	0	0	0
b = 3	0	15	0	0	0	0	15
b = 4	0	0	40	0	0	0	40
b = 5	0	0	90	45	0	0	135
b = 6	0	0	0	240	24	0	264
b = 7	0	0	0	90	170	5	265
Total	1	15	130	375	194	5	720

Tabela A.6: Distribuição das permutações de comprimento 6. O valor b representa o número de pontos de quebra de prefixo e d a distância de transposição de prefixo.

n = 7	d = 0	d = 1	d = 2	d = 3	d = 4	d = 5	d = 6	Total
b = 1	1	0	0	0	0	0	0	1
b = 2	0	0	0	0	0	0	0	0
b = 3	0	21	0	0	0	0	0	21
b = 4	0	0	70	0	0	0	0	70
b = 5	0	0	210	105	0	0	0	315
b = 6	0	0	0	840	84	0	0	924
b = 7	0	0	0	630	1190	35	0	1855
b = 8	0	0	0	0	1324	527	3	1854
Total	1	21	280	1575	2598	562	3	5040

Tabela A.7: Distribuição das permutações de comprimento 7. O valor b representa o número de pontos de quebra de prefixo e d a distância de transposição de prefixo.

n = 8	d = 0	d = 1	d = 2	d = 3	d = 4	d = 5	d = 6	Total
b = 1	1	0	0	0	0	0	0	1
b = 2	0	0	0	0	0	0	0	0
b = 3	0	28	0	0	0	0	0	28
b = 4	0	0	112	0	0	0	0	112
b = 5	0	0	420	210	0	0	0	630
b = 6	0	0	0	2240	224	0	0	2464
b = 7	0	0	0	2520	4760	140	0	7420
b = 8	0	0	0	0	10592	4216	24	14832
b = 9	0	0	0	0	2520	11176	1137	14833
Total	1	28	532	4970	18096	15532	1161	40320

Tabela A.8: Distribuição das permutações de comprimento 8. O valor b representa o número de pontos de quebra de prefixo e d a distância de transposição de prefixo.

n = 9	d = 0	d = 1	d = 2	d = 3	d = 4	d = 5	d = 6	d = 7	Total
b = 1	1	0	0	0	0	0	0	0	1
b = 2	0	0	0	0	0	0	0	0	0
b = 3	0	36	0	0	0	0	0	0	36
b = 4	0	0	168	0	0	0	0	0	168
b = 5	0	0	756	378	0	0	0	0	1134
b = 6	0	0	0	5040	504	0	0	0	5544
b = 7	0	0	0	7560	14280	420	0	0	22260
b = 8	0	0	0	0	47664	18972	108	0	66744
b = 9	0	0	0	0	22680	100584	10233	0	133497
b = 10	0	0	0	0	0	68410	63842	1244	133496
Total	1	36	924	12978	85128	188386	74183	1244	362880

Tabela A.9: Distribuição das permutações de comprimento 9. O valor b representa o número de pontos de quebra de prefixo e d a distância de transposição de prefixo.

n = 10	d = 0	d = 1	d = 2	d = 3	d = 4	d = 5	d = 6	d = 7	d = 8	Total
b = 1	1	0	0	0	0	0	0	0	0	1
b = 2	0	0	0	0	0	0	0	0	0	0
b = 3	0	45	0	0	0	0	0	0	0	45
b = 4	0	0	240	0	0	0	0	0	0	240
b = 5	0	0	1260	630	0	0	0	0	0	1890
b = 6	0	0	0	10080	1008	0	0	0	0	11088
b = 7	0	0	0	18900	35700	1050	0	0	0	55650
b = 8	0	0	0	0	158880	63240	360	0	0	222480
b = 9	0	0	0	0	113400	502920	51165	0	0	667485
b = 10	0	0	0	0	0	684100	638420	12440	0	1334960
b = 11	0	0	0	0	0	113400	989244	231990	327	1334961
Total	1	45	1500	29610	308988	1364710	1679189	244430	327	3628800

Tabela A.10: Distribuição das permutações de comprimento 10. O valor b representa o número de pontos de quebra de prefixo e d a distância de transposição de prefixo.

n = 11	d = 0	d = 1	d = 2	d = 3	d = 4	d = 5
b = 1	1	0	0	0	0	0
b = 2	0	0	0	0	0	0
b = 3	0	55	0	0	0	0
b = 4	0	0	330	0	0	0
b = 5	0	0	1980	990	0	0
b = 6	0	0	0	18480	1848	0
b = 7	0	0	0	41580	78540	2310
b = 8	0	0	0	0	436920	173910
b = 9	0	0	0	0	415800	1844040
b = 10	0	0	0	0	0	3762550
b = 11	0	0	0	0	0	1247400
b = 12	0	0	0	0	0	0
Total	1	55	2310	61050	933108	7030210

Tabela A.11: Distribuição das permutações de comprimento 11. O valor b representa o número de pontos de quebra de prefixo e d a distância de transposição de prefixo.

n = 11	d = 6	d = 7	d = 8	d = 9	Total
b = 1	0	0	0	0	1
b = 2	0	0	0	0	0
b = 3	0	0	0	0	55
b = 4	0	0	0	0	330
b = 5	0	0	0	0	2970
b = 6	0	0	0	0	20328
b = 7	0	0	0	0	122430
b = 8	990	0	0	0	611820
b = 9	187605	0	0	0	2447445
b = 10	3511310	68420	0	0	7342280
b = 11	10881684	2551890	3597	0	14684571
b = 12	5131953	9139366	413248	3	14684570
Total	19713542	11759676	416845	3	39916800

Tabela A.12: Distribuição das permutações de comprimento 11 (continuação). O valor b representa o número de pontos de quebra de prefixo e d a distância de transposição de prefixo.

Referências Bibliográficas

[1] 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. *Journal of Computational Biology*, 8(5):483–491, 2001.

[2] V. Bafna and P. A. Pevzner. Sorting by reversals: Genome rearrangements in plant organelles and evolutionary history of X chromosome. *Molecular Biology and Evolution*, 12(2):239–246, 1995.

[3] V. Bafna and P. A. Pevzner. Genome rearrangements and sorting by reversals. *SIAM Journal on Computing*, 25(2):272–289, 1996.

[4] V. Bafna and P. A. Pevzner. Sorting by transpositions. *SIAM Journal on Discrete Mathematics*, 11(2):224–240, May 1998.

[5] A. Bergeron. A very elementary presentation of the Hannenhalli-Pevzner theory. In *Proceedings of the 12th Annual Symposium of the Combinatorial Pattern Matching (CPM'2001)*, volume 2089 of *Lecture Notes in Computer Science*, pages 106–117, Berlin, Germany, September 2001. Springer-Verlag.

[6] P. Berman, S. Hannenhalli, and M. Karpinski. 1.375-approximation algorithm for sorting by reversals. In *Proceedings of the 10th European Symposium on Algorithms (ESA'2002)*, Lecture Notes in Computer Science, Rome, Italy, September 2002. Springer.

[7] A. Caprara. Sorting by reversals is difficult. In *RECOMB '97: Proceedings of the first annual international conference on Computational molecular biology*, pages 75–83. ACM Press, 1997.

[8] 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, San Francisco, USA, January 1998.

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

[10] Z. Dias and J. Meidanis. Sorting by prefix transpositions. In A. H. F. Laender and A. L. Oliveira, editors, *Proceedings of the String Processing and Information Retrieval (SPIRE'2002)*, number 2476 in Lecture Notes in Computer Science, pages 65–76, Lisboa, Portugal, September 2002. Springer-Verlag, Berlin.

[11] N. Eriksen. $(1+\varepsilon)$-Approximation of sorting by reversals and transpositions. *Theoretical Computer Science*, 289:517–529, 2002.

[12] H. Eriksson, K. Eriksson, J. Karlander, L. Svensson, and J. Wästlund. Sorting a bridge hand. *Discrete Mathematics*, 241(241):289–300, 2001.

[13] V. J. Fortuna and J. Meidanis. Sorting the reverse permutation by prefix transpositions. Technical Report IC-04-04, Institute of Computing - University of Campinas, April 2004.

[14] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics; Second Edition.* Addison-Wesley, 1997.

[15] Q.-P. Gu, S. Peng, and H. Sudborough. A 2-approximation algorithm for genome rearrangements by reversals and transpositions. *Theoretical Computer Science*, 210(2):327–339, January 1999.

[16] S. A. Guyer, L. S. Heath, and J. P. Vergara. Subsequence and run heuristics for sorting by transpositions. In *Proceedings of the 4th DIMACS International Algorithm Implementation Challenge*, August 1995.

[17] S. Hannenhalli and P. A. Pevzner. Transforming cabbage into turnip: Polynomial algorithm for sorting signed permutations by reversals. *Journal of the ACM*, 46(1):1–27, January 1999.

[18] T. Hartman and R. Shamir. A simpler and faster 1.5-approximation algorithm for sorting by transpositions. Uma versão preliminar deste trabalho foi publicado nos anais do congresso *Combinatorial Pattern Matching* de 2003 (CPM'03), 2004.

[19] H. Kaplan, R. Shamir, and R. E. Tarjan. Faster and simpler algorithm for sorting signed permutations by reversals. *SIAM Journal on Computing*, 29(3):880–892, January 2000.

[20] G.-H. Lin and G. Xue. Signed genome rearrangement by reversals and transpositions: models and approximations. *Theor. Comput. Sci.*, 259(1-2):513–531, 2001.

[21] J. Meidanis and Z. Dias. An alternative algebraic formalism for genome rearrangements. In D. Sankoff and J. H. Nadeau, editors, *Proceedings of the Gene Order Dynamics, Comparative Maps and Multigene Families (DCAF'2000)*, Le Chantecler, Canada, September 2000.

[22] J. Meidanis, M. E. M. T. Walter, and Z. Dias. Transposition distance between a permutation and its reverse. In R. Baeza-Yates, editor, *Proceedings of the 4th South American Workshop on String Processing (WSP'97)*, pages 70–79, Valparaiso, Chile, 1997. Carleton University Press.

[23] J. Meidanis, M. E. M. T. Walter, and Z. Dias. Reversal distance of signed circular chromosomes. Technical Report IC-00-23, Institute of Computing - University of Campinas, December 2000.

[24] J. D. Palmer and L. A. Herbon. Plant mitochondrial DNA evolves rapidly in structure, but slowly in sequence. *Journal of Molecular Evolution*, 27:87–97, 1988.

[25] P. A. Pevzner. *Computational Molecular Biology: An Algorithmic Approach.* The MIT Press, 2000.

[26] J. C. Setubal and J. Meidanis. *Introduction to Computional Molecular Biology.* PWS Publishing Company, 1997.

[27] A. Gullberg U. Arnason, X. Xu. Comparison between the complete mitochondrial DNA sequences of homo and the common chimpanzee based on nonchimeric sequences. *Journal of Molecular Evolution*, 42(2):145 – 152, 1996.

[28] M. E. M. T. Walter, Z. Dias, and J. Meidanis. Reversal and transposition distance of linear chromosomes. In *Proceedings of the String Processing and Information Retrieval (SPIRE'98)*, 1998.

[29] M. E. M. T. Walter, Z. Dias, and J. Meidanis. A new approach for approximating the transposition distance. In *Proceedings of the String Processing and Information Retrieval (SPIRE'2000)*, September 2000.