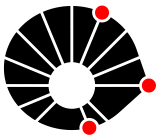

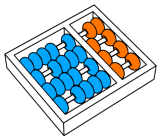

Ordenação de Permutações por Operações Determinísticas

Matheus Rotta *Gabriel Siqueira* *Zanoni Dias*

Relatório Técnico - IC-PFG-21-06
Projeto Final de Graduação
2021 - Julho

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

Ordenação de Permutações por Operações Determinísticas

Matheus Rotta Gabriel Siqueira Zanoni Dias

Resumo

O problema de ordenação apresenta diversas aplicações, por exemplo, o problema de rearranjo de genomas pode ser formulado como o de ordenação de permutações quando não há repetição de genes no modelo. Neste trabalho, exploramos as operações de troca, reversão e transposição. Essas operações podem ou não ser determinísticas, o que significa que um ou mais elementos devem ser posicionados corretamente. Quando falamos em uma operação parcialmente determinística, nos referimos ao fato de pelo menos um dos elementos envolvidos ir para a posição correta; já no caso de uma operação totalmente determinística, todos os elementos envolvidos na operação são posicionados corretamente. Primeiro, estudamos a versão parcialmente determinística, e depois a totalmente determinística para cada operação. No caso da troca, geramos algoritmos ou usamos algoritmos clássicos que conseguiam seguir a restrição de determinismo e manter o número mínimo de operações. Já no caso de reversão e transposição, foram desenvolvidas heurísticas que buscavam ordenar a permutação numa quantidade de operações próxima da mínima, seguindo as restrições de determinismo parcial. Para todas as operações com determinismo total descrevemos as permutações que podem ser ordenadas. Como esperado, ao se impor que as operações sejam parcialmente determinísticas, o número de operações necessárias para ordenar a permutação é maior já que há menos operações disponíveis. Além disso, constatamos que esse aumento é maior no caso da reversão do que na transposição. Foram necessárias em média 1.6 vezes mais operações de reversão e 1.2 vezes mais operações de transposição para ordenar as permutações ao forçar o determinismo.

1 Introdução

O problema de ordenação [12] é um problema computacional muito estudado e fundamental em várias aplicações computacionais. Em uma de suas formulações mais simples o problema consiste em ordenar uma lista de elementos que possuem uma relação de ordem entre eles. Normalmente, os elementos a serem ordenados são inteiros, ou possuem uma representação como inteiros, e a ordem considerada é a ordem natural dos inteiros.

Existem aplicações onde o problema de ordenação apresenta outras restrições que podem torná-lo mais simples ou mais complexo. Além disso, nem sempre estamos interessados apenas em ordenar a lista de elementos. Em alguns casos, o objetivo é encontrar uma sequência minima de operações que ordene essa lista. Por exemplo, uma variação do problema de ordenação pode considerar apenas uma lista sem elementos repetidos, que pode ser representada por uma permutação, e buscar minimizar o número de operações de troca de elementos que ordene essa permutação.

Dentre as variações do problema de ordenação, algumas possuem aplicações em problemas encontrados em outras áreas. No campo de biologia computacional, temos os problemas de distância de rearranjo [8] que consistem em encontrar o menor número de eventos de rearranjo, uma forma de mutação, que transformam um genoma em outro. Quando consideramos que os dois genomas têm os mesmos genes e nenhum gene possui mais de uma cópia, esses problemas podem ser modelados como problemas de ordenação de permutações por um número mínimo de operações específicas. Dentre as operações que são consideradas temos a reversão [13], que inverte uma sequência continua de genes, e a transposição [2], que troca a ordem de duas sequências continuas consecutivas de genes. Os problemas de ordenar permutações por uma dessas operações pertencem à classe NP-difícil [5, 6] e o melhor fator de aproximação conhecido para eles é de 1.375 [4, 7].

Um outra possível variação de problemas de ordenação que consideram operações específicas, é restringir o conjunto de operações que podem ser aplicadas ao requerer que cada operação posicione elementos corretamente. Operações com essa restrição são chamados de operações determinísticas. Por exemplo, Aigner e West [1] estudaram a versão determinística do problema de ordenar uma permutação por inserção do primeiro elemento. Na versão original, esse problema consiste em ordenar uma permutação por uma operação que remove o primeiro elemento e o adiciona em qualquer posição. Na versão determinística o elemento removido tem que ser inserido na posição correta. Outro exemplo, é o problema que consiste em ordenar uma permutação por reversão de prefixo [10], que também teve uma versão determinística explorada [3].

Neste trabalho estudamos a inclusão da restrição de operações determinísticas em problemas de ordenação envolvendo as operações de troca de elementos, reversão e transposição. Na Seção 2 apresentamos algumas definições sobre permutações. Nas seções de 3 até 5, abordamos cada uma das operações estudadas. Na Seção 6 mostramos os resultados dos experimentos práticos realizados e, por fim, a Seção 7 conclui o trabalho.

2 Permutações

Nesta seção apresentamos algumas definições gerais sobre permutações que serão utilizadas nas próximas seções.

Uma *permutação* π é uma bijeção sobre o conjunto $\{1, 2, \ldots, n\}$ e a imagem de $i \in \{1, 2, \ldots, n\}$ por π é denotada por π_i. A *permutação inversa* de π, denotada por π^{-1}, é tal que $\pi^{-1}_{\pi_i} = i$, ou seja, o valor π^{-1}_x nos indica a posição de x em π.

Exemplo 1. Uma permutação π e sua inversa π^{-1}.

$$\pi = (2\ 3\ 1\ 7\ 6\ 4\ 5)$$
$$\pi^{-1} = (3\ 1\ 2\ 6\ 7\ 5\ 4),\ \pi_1 = 2,\ \pi^{-1}_2 = 1$$

Definimos como a *permutação identidade* ι_n a permutação tal que $\iota_i = i$ para todo $i \in \{1, 2, \ldots, n\}$, ou seja, a permutação já ordenada. Nos problemas de ordenação que estudamos, o objetivo é transformar uma permutação qualquer na permutação identidade através de operações específicas.

Um elemento π_i de uma permutação π é dito *auto inverso* se $\pi_i = \pi_i^{-1}$, ou seja $\pi_{\pi_i} = i$. Uma permutação π é dita *auto inversa* se $\pi = \pi^{-1}$, ou seja, todos seus elementos são auto inversos. Note que a permutação identidade é um caso particular de permutação auto inversa. Conforme apresentado em [9], o número I_n de permutações auto inversas de tamanho n é dado por:

$$I_n = \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \frac{n!}{(n-2k)!2^k k!}$$

Um elemento auto inverso π_i satisfaz uma das seguintes condições:

1. $\pi_i = i$, nesse caso dizemos que π_i é um elemento *fixo* de π;
2. existe outro elemento π_j, tal que $\pi_i = j$ e $\pi_j = i$.

Denotaremos por $fix(\pi)$ o número de elementos fixos de uma permutação π. Além disso, denotamos por $ainv(\pi)$ o número de elementos auto inversos não fixos de π. A permutação ι_n é a única permutação de tamanho n com $fix(\iota_n) = n$.

Exemplo 2. Uma permutação auto inversa π. Os elementos fixos de π são 1, 5 e 6.

$$\pi = \pi^{-1} = (1\ 3\ 2\ 7\ 5\ 6\ 4)$$
$$fix(\pi) = 3, \quad ainv(\pi) = 4$$
$$\pi_2 = 3, \quad \pi_3 = 2, \quad \pi_4 = 7, \quad \pi_7 = 4$$

3 Trocas

Nesta seção tratamos sobre as variações determinísticas da operação de troca. Dada uma permutação $\pi = (\pi_1 \dots \pi_{i-1}\underline{\pi_i}\pi_{i+1} \dots \pi_{j-1}\underline{\pi_j}\pi_{j+1} \dots \pi_n)$, uma *troca* $t(i,j)$, com $1 \leq i \leq n$, $1 \leq j \leq n$ e $i \neq j$, é uma operação que troca os elementos i e j de posição, resultando na permutação $\pi \circ t(i,j) = (\pi_1 \dots \pi_{i-1}\underline{\pi_j}\pi_{i+1} \dots \pi_{j-1}\underline{\pi_i}\pi_{j+1} \dots \pi_n)$.

Uma troca é dita *parcialmente determinística* se afeta um elemento π_k (ou seja, $i = k$ ou $j = k$) tal que $\pi'_k = k$ ou $\pi'_{\pi_k} = \pi_k$, onde $\pi' = \pi \circ t(i,j)$. Além disso, uma troca é dita *totalmente determinística* se afeta um elemento π_k tal que $\pi'_k = k$ e $\pi'_{\pi_k} = \pi_k$, onde $\pi' = \pi \circ t(i,j)$. Essencialmente, uma troca parcialmente determinística posiciona corretamente um dos elementos afetados pela troca enquanto uma troca totalmente determinística posiciona corretamente os dois elementos afetados pela troca.

Exemplo 3. Três trocas $t(1,6)$, $t(4,5)$ e $t(2,4)$ sendo aplicadas em uma permutação π. A troca $t(1,6)$ é parcialmente determinística, pois $\pi_6 = 1$, a troca $t(4,5)$ é totalmente determinística, pois $\pi_4 = 5$ e $\pi_5 = 4$, e a troca $t(2,4)$ não é determinística.

$$\pi = (2\ 3\ 6\ 5\ 4\ 1)$$
$$\pi \circ t(1,6) = (\underline{1}\ 3\ 6\ 5\ 4\ \underline{2})$$
$$\pi \circ t(4,5) = (2\ 3\ 6\ \underline{4}\ \underline{5}\ 1)$$
$$\pi \circ t(2,4) = (2\ \underline{5}\ 6\ \underline{3}\ 4\ 1)$$

3.1 Trocas Parcialmente Determinísticas

Vamos começar tratando sobre trocas parcialmente determinísticas. Começamos com alguns resultados que relacionam trocas parcialmente determinísticas com elementos fixos e auto inversos.

Lema 1. *Dada uma permutação π. Nenhuma troca parcialmente determinística afeta um elemento fixo de π.*

Demonstração. Considere uma troca parcialmente determinística $t(i,j)$ que afete um elemento fixo k de uma permutação π e seja $\pi' = \pi \circ t(i,j)$. Devemos ter $\pi'_k = k$ ou $\pi'_{\pi_k} = \pi_k$. Caso $\pi'_k = k$, como $\pi_k = k$, temos $\pi'_k = \pi_k$. Chegamos a uma contradição, pois toda troca que afeta a posição k muda o valor de π_k. De forma similar se $\pi'_{\pi_k} = \pi_k$ temos $\pi'_k = \pi'_{\pi_k} = \pi_k$ e chegamos a mesma contradição. □

Uma consequência direta do Lema 1 é que nenhuma troca parcialmente determinística pode ser aplicada em uma permutação ordenada.

Lema 2. *Dada uma permutação π e um elemento auto inverso k de π, temos as seguintes implicações: (i) se $k < \pi_k$, a troca $t(k, \pi_k)$ é a única troca parcialmente determinística que pode afetar k; (ii) se $k > \pi_k$, a troca $t(k, \pi_k)$ é a única troca parcialmente determinística que pode afetar k; (iii) se $k = \pi_k$, nenhuma troca parcialmente determinística pode afetar k.*

Demonstração. O item (iii) é uma consequencia direta do Lema 1. Para os itens (i) e (ii), caso uma troca $t(i,j)$ afete o elemento k, devemos ter $\pi'_k = k$ ou $\pi'_{\pi_k} = \pi_k$, em ambos os casos π_k também é afetado. Logo, se $k < \pi_k$ temos $t(i,j) = t(k, \pi_k)$, e se $k > \pi_k$, temos $t(i,j) = t(\pi_k, k)$. □

Denotaremos por $\Delta fix(\pi, t(i,j)) = fix(\pi \circ t(i,j)) - fix(\pi)$ a variação do valor de fix após a aplicação de uma troca $t(i,j)$.

Lema 3. *Dada uma permutação π, para toda troca parcialmente determinística $t(i,j)$ temos $\Delta fix(\pi, t(i,j)) \geq 1$. Além disso, se $\Delta fix(\pi, t(i,j)) = 2$ então i e j são ambos auto inversos.*

Demonstração. Considere uma troca parcialmente determinística $t(i,j)$ que afete k e seja $\pi' = \pi \circ t(i,j)$. Devemos ter $\pi'_k = k$ ou $\pi'_{\pi_k} = \pi_k$. Caso $\pi'_{\pi_k} = \pi_k$, π_k não estava fixo antes e agora está. Caso $\pi'_k = k$, k não estava fixo antes e agora está. Como a troca afeta apenas k e π_k, temos no máximo dois novos elementos fixos e, pelo Lema 1, nenhum elemento fixo foi afetado.

Agora, suponha sem perda de generalidade que $i = k$. Se dois elementos fixos foram adicionados, devemos ter $\pi'_k = k$, o que implica que $\pi_j = k$, e $\pi'_{\pi_k} = \pi_k$, o que implica que $\pi_k = j$, portanto $\pi_{\pi_i} = \pi_{\pi_k} = k = i$ e $\pi_{\pi_j} = \pi_{\pi_{\pi_k}} = \pi_k = j$. Ou seja, i e j são auto inversos. □

Teorema 1. *Qualquer sequência de trocas parcialmente determinísticas que ordene uma dada permutação π de tamanho n utiliza exatamente $n - fix(\pi) - \frac{ainv(\pi)}{2}$ trocas.*

Demonstração. Como $fix(\iota_n) = n$, sabemos que qualquer sequência que ordene π deve aumentar o número de elementos fixos até n. Como já temos $fix(\pi)$ elementos fixos precisamos fixar mais $n - fix(\pi)$ elementos. Pelos lemas 2 e 3 as trocas parcialmente determinísticas que aumentam em dois o número de elementos fixos devem estar presente em qualquer sequência que ordene π e cada uma dessas trocas fixa dois elementos auto inversos, que necessariamente não estão fixos. Note que temos $\frac{ainv(\pi)}{2}$ trocas dessa forma e elas fixam $ainv(\pi)$ elementos. Além disso, pelo Lema 3, as demais trocas parcialmente determinísticas, que não afetam elementos auto inversos, aumentam em exatamente um o número de elementos fixos, portanto devemos ter $n - fix(\pi) - ainv(\pi)$ trocas dessa forma. No total $n - fix(\pi) - \frac{ainv(\pi)}{2}$ trocas parcialmente determinísticas são necessárias e suficientes para ordenar π. □

Agora vamos provar que o Selection Sort (Algoritmo 1) utiliza apenas trocas parcialmente determinísticas, ou seja, todas as trocas realizadas pelo algoritmo colocam pelo menos um elemento em sua posição correta.

Algoritmo 1: Selection Sort

Entrada: Permutação π de tamanho n
Saída: Sequência de trocas parcialmente determinísticas capazes de ordenar π

```
1  início
2      S ← Sequência vazia
3      para i de 1 até n − 1 faça
4          minIndex ← i
5          minValue ← π_i
6          para j de i+1 até n faça
7              se π_minIndex > π_j então
8                  minIndex ← j
9                  minValue ← π_j
10         se i ≠ minIndex então
11             Adicione a troca t(i, j) em S
12             π ← π ∘ t(i, j)
13     retorna S
```

Teorema 2. *Cada troca realizada pelo algoritmo Selection Sort é uma troca parcialmente determinística.*

Demonstração. Ao longo do algoritmo, o Selection Sort separa em dois a permutação a ser ordenada. No caso de uma permutação π de tamanho n, suponha que $i = k$, $1 \leq k < n$. Como a parte da permutação já ordenada vai de 1 até $k - 1$, o menor elemento da parte não ordenada a ser selecionado no laço das linhas 4 a 9 será k, e será trocado com qualquer elemento que esteja na k-ésima posição (linha 12). Deste modo, a troca na linha 12 sempre colocará pelo menos um elemento na posição correta, que é o elemento k. □

3.2 Trocas Totalmente Determinísticas

Vamos agora verificar que as permutações auto inversas são as únicas que podem ser ordenadas por trocas totalmente determinísticas.

Teorema 3. *Uma permutação π de tamanho n é ordenável por trocas totalmente determinísticas somente se é uma permutação auto inversa.*

Demonstração. Vamos provar por indução no tamanho n da permutação.

Base: para $n = 1$, $\pi = (1)$ é a única permutação e $\pi_1 = 1$.

Hipótese de Indução: para $1 < k < n$, uma permutação π de tamanho k é ordenável por trocas totalmente determinísticas somente se é uma permutação auto inversa.

Passo: dada uma permutação π de tamanho $n > 1$, temos dois casos possíveis:

- $\pi_1 = 1$, então o elemento 1 é fixo e aplicamos a hipótese de indução para garantir que os outros $n - 1$ elementos são auto inversos.
- $\pi_1 = j \neq 1$, podemos provar por absurdo: se $\pi_j \neq 1$, então não poderemos corrigir o elemento na primeira posição, pois precisaríamos levar o elemento j à sua posição correta j, mas como $\pi_j \neq 1$, não podemos realizar a troca. Também não é possível trocar o elemento na posição j com um terceiro elemento em uma posição k tal que $\pi_k = 1$ pois o valor 1 teria que ir para a posição 1, mas na verdade vai para a posição $j \neq 1$, o que também é impossível. Portanto, se $\pi_1 = j$, então $\pi_j = 1$. Considere agora a permutação π' correspondente a π sem os elementos j e π_j e renumerada para ir de 1 até $n-2$. Podemos ordenar π por trocas totalmente determinísticas se e somente se pudermos ordenar π' por trocas totalmente determinísticas. Pela hipótese de indução π' deve ser auto inversa, o que garante que π deve ser auto inversa. □

O Algoritmo 2 mostra como ordenar as permutações que são ordenáveis com apenas trocas totalmente determinísticas.

Exemplo 4. A permutação $\pi = (3\ 1\ 2\ 4)$ não é ordenável por trocas totalmente determinísticas. O algoritmo tenta posicionar corretamente o elemento 1 realizando a troca $t(1, 3)$, mas não é possível pois $\pi_3 \neq 1$.

Exemplo 5. A permutação $\pi = (3\ 2\ 1\ 7\ 5\ 6\ 4)$ é ordenável pela seguinte sequência de trocas totalmente determinísticas.

$$\begin{aligned} \pi &= (3\ 2\ 1\ 7\ 5\ 6\ 4) \\ \pi' = \pi \circ t(1,3) &= (\underline{1}\ 2\ \underline{3}\ 7\ 5\ 6\ 4) \\ \iota_n = \pi' \circ t(4,7) &= (1\ 2\ 3\ \underline{4}\ 5\ 6\ \underline{7}) \end{aligned}$$

Algoritmo 2: Ordenação por trocas totalmente determinísticas

Entrada: Permutação π de tamanho n
Saída: Sequência de trocas totalmente determinísticas capazes de ordenar π

```
1  início
2      S ← Sequência vazia
3      para i de 1 a n faça
4          se π_i ≠ i então
5              se π_{π_i} ≠ i então
6                  Erro: π não é ordenável por trocas totalmente determinísticas
7              senão
8                  Adiciona a troca t(i, π_i) em S
9                  π ← π ∘ t(i, π_i)
10     retorna S
```

4 Reversões

Nesta seção, exploramos a operação de reversão. Para simplificar as definições, assumiremos que cada permutação π é estendida com a adição de dois novos elementos $\pi_0 = 0$ e $\pi_{n+1} = n + 1$, onde n é o tamanho de π.

Exemplo 6. Uma permutação π estendida com a adição dos elementos $\pi_0 = 0$ e $\pi_7 = 7$ resultando na permutação π'.

$$\pi = \quad (2\ 3\ 6\ 5\ 4\ 1)$$
$$\pi' = (\mathbf{0}\ 2\ 3\ 6\ 5\ 4\ 1\ \mathbf{7})$$

Dada uma permutação $\pi = (\pi_0 \dots \pi_{i-1}\underline{\pi_i \dots \pi_j}\pi_{j+1} \dots \pi_{n+1})$, uma *reversão* $\rho(i, j)$, com $1 \leq i < j \leq n$, é uma operação que inverte a sequência de elementos entre i e j resultando na permutação $\pi \circ \rho(i, j) = (\pi_0 \dots \pi_{i-1}\underline{\pi_j \dots \pi_i}\pi_{j+1} \dots \pi_{n+1})$. Uma reversão $\rho(i, j)$ é dita *parcialmente determinística* se $\pi_i = j$ ou $\pi_j = i$, ou seja, um dos extremos da sequência invertida é posicionado corretamente. Além disso, uma reversão $\rho(i, j)$ é dita *totalmente determinística* se $\pi_k = j - k + i, \forall i \leq k \leq j$, ou seja, todos os elementos da sequência invertida são posicionados corretamente.

Exemplo 7. Três reversões $\rho(1, 7)$, $\rho(4, 6)$ e $\rho(2, 5)$ sendo aplicadas em uma permutação π. A reversão $\rho(1, 7)$ é parcialmente determinística, pois $\pi_7 = 1$, a reversão $\rho(4, 6)$ é totalmente determinística, pois $\pi_4 = 6$, $\pi_5 = 5$ e $\pi_6 = 4$, e a reversão $\rho(2, 5)$ não é determinística.

$$\pi = (0\ 3\ 2\ 7\ 6\ 5\ 4\ 1\ 8)$$
$$\pi \circ \rho(1, 7) = (0\ \underline{1\ 4\ 5\ 6\ 7\ 2\ 3}\ 8)$$
$$\pi \circ \rho(4, 6) = (0\ 3\ 2\ 7\ \underline{4\ 5\ 6}\ 1\ 8)$$
$$\pi \circ \rho(2, 5) = (0\ 3\ \underline{5\ 6\ 7\ 2}\ 4\ 1\ 8)$$

Quando queremos ordenar uma permutação usando o número mínimo de reversões, temos o problema de Ordenação de Permutações por Reversão. Quando nos restringimos às reversões determinísticas, temos os problemas de Ordenação de Permutações por Reversão Parcialmente Determinística e Ordenação de Permutações por Reversão Totalmente Determinística.

Precisamos definir mais alguns conceitos para lidar com reversões. Dada uma permutação π, um par de elementos consecutivos (π_i, π_{i+1}) é um *breakpoint de reversão* se $|\pi_{i+1} - \pi_i| \neq 1$. Denotamos o número de breakpoints de reversão de uma permutação π por $b_\rho(\pi)$. Dizemos que uma reversão $\rho(i,j)$ remove x breakpoints de reversão de uma permutação π se $b_\rho(\pi \circ \rho(i,j)) - b_\rho(\pi) = x$. Note que a permutação identidade ι_n é a única permutação de tamanho n para a qual b_ρ é 0. Uma sequência maximal de elementos de π sem breakpoints de reversão é chamada de uma *strip de reversão* de π. Uma strip de reversão é *crescente* se ela possui apenas um elemento ou seus elementos estão em ordem crescente, caso contrário a strip é *decrescente*.

Exemplo 8. Os breakpoints de reversão de uma permutação π. Cada breakpoint é indicado por um ponto (•). As strips de reversão crescentes de π são (0), (2 3), (1) e (7); e (6 5 4) é uma strip de reversão decrescente.

$$\pi = (0 \ \bullet \ 2\ 3 \ \bullet \ 6\ 5\ 4 \ \bullet \ 1 \ \bullet \ 7)$$

4.1 Reversões Parcialmente Determinísticas

A seguir, propomos um algoritmo para ordenar permutações por reversões parcialmente determinísticas. O algoritmo se baseia no seguinte lema.

Lema 4 (Kececioglu e Sankoff [11])**.** *Uma reversão $\rho(i,j)$ aplicada em uma permutação π pode remover no máximo 2 breakpoints de reversão.*

Como consequência do Lema 4, temos que pelo menos $\frac{b_\rho(\pi)}{2}$ reversões são necessárias para ordenar uma permutação π. De fato, Kececioglu e Sankoff [11] demostraram que é possível remover em média pelo menos um breakpoint por reversão, o que garante um fator de aproximação 2 para o problema de Ordenação de Permutações por Reversão. Essa mesma garantia não existe para o problema de Ordenação por Reversão Parcialmente Determinística, mas utilizamos o conceito de breakpoints para produzir uma heurística gulosa para o problema.

O Algoritmo 3 apresenta um pseudocódigo do algoritmo desenvolvido. A ideia é buscar por reversões parcialmente determinísticas que removam o maior número de breakpoints. Caso nenhuma reversão seja capaz de remover breakpoints, seja π_i o primeiro elemento fora de ordem em π, aplicamos a reversão $\rho(i,j)$, tal que $\pi_j = i$.

Teorema 4. *Dada uma permutação π, o Algoritmo 3 para e encontra uma sequência que ordena π por reversões parcialmente determinísticas.*

Demonstração. Considere que estamos em uma interação qualquer do algoritmo e seja π_i o primeiro elemento fora de ordem em π. Temos que nenhum elemento π_k, $k < i$ é afetado

Algoritmo 3: Ordenação por reversões parcialmente determinísticas

Entrada: Permutação π
Saída: Sequência de reversões parcialmente determinísticas capazes de ordenar π

```
1  início
2      S ← sequência vazia
3      enquanto π não estiver ordenada faça
4          se existe uma reversão determinística ρ(i, j) que remove 2 breakpoints então
5              Adicione ρ(i, j) em S
6              π ← π ∘ ρ(i, j)
7          senão se existe uma reversão determinística ρ(i, j) que remove 1 breakpoint
               então
8              Adicione ρ(i, j) em S
9              π ← π ∘ ρ(i, j)
10         senão
11             Seja π_i o primeiro elemento fora de ordem em π e j, tal que π_j = i
12             Adicione ρ(i, j) em S
13             π ← π ∘ ρ(i, j)
14     retorna S
```

pelas operações realizadas pelo algoritmo, pois essa operação aumentaria o número de breakpoints. Além disso, a cada passo a reversão aplicada diminui o número de breakpoints de π ou posiciona corretamente o primeiro elemento não ordenado de π. Como o número de breakpoints de π só aumenta no segundo caso, quando o número de elementos a serem ordenados diminui, eventualmente a permutação estará ordenada.

Além disso, todas as operações que removem breakpoints são escolhidas apenas quando são determinísticas e a última operação é determinística, pois $\pi_j = i$. □

Exemplo 9. A permutação $\pi = (0\ 3\ 2\ 1\ 7\ 5\ 6\ 4\ 8)$ é ordenada pelo algoritmo 3 com a seguinte sequência de reversões parcialmente determinísticas.

$$\begin{aligned}
\pi &= (0\ 3\ 2\ 1\ 6\ 5\ 7\ 4\ 8)\\
\pi' = \pi \circ \rho(6,7) &= (0\ 3\ 2\ 1\ 6\ 5\ \underline{4\ 7}\ 8)\\
\pi'' = \pi' \circ \rho(1,3) &= (0\ \underline{1\ 2\ 3}\ 6\ 5\ 4\ 7\ 8)\\
\iota_n = \pi'' \circ \rho(4,6) &= (0\ 1\ 2\ 3\ \underline{4\ 5\ 6}\ 7\ 8)
\end{aligned}$$

Vamos analisar brevemente a complexidade de tempo do algoritmo 3. Note que em cada interação do laço das linhas 3 até 13 reduzimos em pelo menos 1 o número de breakpoints ou reduzimos o número de elementos a serem ordenados e aumentamos em no máximo 1 o número de de breakpoints. Dessa forma, o laço é executado $O(n)$ vezes. Em cada iteração, como consideramos operações determinísticas, só temos que verificar as operações que posicionam corretamente cada elemento. Além disso, podemos verificar o número de breakpoits removidos por cada operação em tempo $O(1)$ se mantivermos um vetor representando a

permutação π e um vetor representando a permutação inversa π^{-1}. Portanto, encontrar a operação a ser aplicada leva tempo $O(n)$. Além disso, aplicar a operação encontrada para atualizar a nossa representação também leva tempo $O(n)$. No total a complexidade do algoritmo é $O(n^2)$.

4.2 Reversões Totalmente Determinísticas

Vamos descrever quais permutações podem ser ordenadas por reversões totalmente determinísticas utilizando as seguintes definições. Uma strip crescente de uma permutação é uma *strip crescente fixa* se ela tiver a forma $(k\ k+1\ \ldots\ \ell-1\ \ell)$, onde k e ℓ são a posição inicial e final da strip, respectivamente. Uma strip decrescente de uma permutação é uma *strip decrescente fixa* se ela tiver a forma $(\ell\ \ell-1\ \ldots\ k+1\ k)$, onde k e ℓ são a posição inicial e final da strip, respectivamente.

Teorema 5. *Uma permutação π pode ser ordenada por reversões totalmente determinísticas somente se toda strip for uma strip crescente fixa ou uma strip decrescente fixa.*

Demonstração. Por definição, uma reversão totalmente determinística $\rho(i,j)$ só pode ser aplicada em π se a sequência entre as posições i e j é da forma $(j\ j-1\ \ldots\ i+1\ i)$. Note que, essa sequência deve estar contida em uma strip decrescente fixa para que ela esteja na posição esperada. Com esse fato, vamos realizar a prova por indução no número m de strips decrescentes de π.

> Base: se π não tem nenhuma strip decrescente nenhuma reversão totalmente determinística pode ser aplicada e a permutação já deve estar ordenada, ou seja, tem apenas uma strip $(0\ \ldots\ n+1)$ que é crescente fixa.
>
> Hipótese de Indução: toda a permutação π com $m-1, m>0$ strips decrescentes e ordenável com a operação de reversão totalmente determinística é tal que toda strip é crescente fixa ou decrescente fixa.
>
> Passo: dada uma permutação π com $m>0$ strips decrescentes. Sabemos que π deve ter pelo menos uma strip decrescente fixa. Seja $(\ell\ \ell-1\ \ldots\ k+1\ k)$ essa strip e considere a permutação $\pi' = \pi \circ \rho(k,\ell)$ que terá $m-1$ strips decrescentes. Pela hipótese de indução toda strip de π' é crescente fixa ou decrescente fixa. Como toda strip decrescente de π diferente de $(\ell\ \ell-1\ \ldots\ k+1\ k)$ também é strip de π', elas são decrescentes fixas e, como toda strip crescente de π é também strip de π' ou está contida em uma strip de π', elas são crescentes fixas. □

Note que, dada uma permutação π no formato especificado pelo Teorema 5, podemos ordená-la por reversões totalmente determinísticas ao aplicar uma reversão em cada strip decrescente.

Exemplo 10. A permutação $\pi = (0\ 2\ 1\ 3\ 4\ 7\ 6\ 5\ 8)$ é ordenável pela seguinte sequência

de reversões totalmente determinísticas.

$$\begin{aligned}\pi &= (0\ 2\ 1\ 3\ 4\ 7\ 6\ 5\ 8)\\ \pi' = \pi \circ \rho(1,2) &= (0\ \underline{1\ 2}\ 3\ 4\ 7\ 6\ 5\ 8)\\ \iota_n = \pi' \circ \rho(5,7) &= (0\ 1\ 2\ 3\ 4\ \underline{5\ 6\ 7}\ 8)\end{aligned}$$

Descrevemos agora o número R_n de permutações de tamanho n ordenáveis por reversões totalmente determinísticas. Temos os casos particulares $R_1 = 1$ e $R_2 = 2$, onde todas as permutações são ordenáveis. Para $n > 3$, apresentamos a seguir uma relação de recorrência. Consideramos $R_0 = 1$ e separamos as permutações em 3 casos:

- Existe uma permutação identidade ι_n.
- Existem $\sum_{k=2}^{n} R_{n-k}$ permutações que começam com uma strip decrescente fixa. Para cada tamanho k da strip, usamos a recorrência para contar as possibilidades para o restante da permutação. Note que uma strip decrescente fixa deve ter tamanho pelo menos 2.
- Existem $\sum_{k=3}^{n}(k-2)R_{n-k}$ permutações que começam com uma strip crescente fixa seguida de uma strip decrescente fixa. Para cada tamanho k da sequência de duas strips, usamos a recorrência para contar as possibilidades para o restante da permutação e multiplicamos por $(k-2)$, para levar em conta a posição do breakpoint entre as duas strips.

Note que não é necessário considerar o caso de uma strip crescente fixa seguida de outra strip crescente fixa, pois nesse caso temos uma única strip crescente fixa composta pelas duas. Juntando todos os casos temos a seguinte relação de recorrência:

$$R_n = 1 + \sum_{k=2}^{n} R_{n-k} + \sum_{k=3}^{n}(k-2)R_{n-k}$$

5 Transposições

Nesta seção, exploramos a operação de transposição. Assumiremos novamente que cada permutação π de tamanho n é estendida com a adição de $\pi_0 = 0$ e $\pi_{n+1} = n+1$.

Dada uma permutação $\pi = (\pi_0 \dots \pi_{i-1}\underline{\pi_i \dots \pi_{j-1}}\ \underline{\pi_j \dots \pi_{k-1}}\pi_k \dots \pi_{n+1})$ uma *transposição* $\tau(i,j,k)$, com $1 \le i < j < k \le n+1$, é uma operação que troca a sequência de elementos entre i e $j-1$ com a sequência de elementos entre j e $k-1$ resultando na permutação $\pi \circ \tau(i,j,k) = (\pi_0 \dots \pi_{i-1}\underline{\pi_j \dots \pi_{k-1}}\ \underline{\pi_i \dots \pi_{j-1}}\pi_k \dots \pi_{n+1})$. Uma transposição $\tau(i,j,k)$ é dita *parcialmente determinística* se $\pi_i = k-j+i$ ou $\pi_j = i$, ou seja, pelo menos um dos elementos iniciais das sequências afetadas é posicionado corretamente. Além disso, uma transposição $\tau(i,j,k)$ é dita *totalmente determinística* se $\pi_\ell = k-j+\ell, \forall i \le \ell < j$ e $\pi_\ell = \ell - j + i, \forall j \le \ell < k$, ou seja, todos os elementos das sequências afetadas são posicionados corretamente.

Exemplo 11. Três transposições $\tau(1,4,5)$, $\tau(4,5,7)$ e $\tau(1,3,6)$ sendo aplicadas em uma permutação π. A transposição $\tau(1,4,5)$ é parcialmente determinística, pois $\pi_1 = 2 = 5-4+1$, a transposição $\tau(4,5,7)$ é totalmente determinística, pois $\pi_4 = 6 = 7-5+4$ e $\pi_5 = 4$, e a transposição $\tau(1,3,6)$ não é determinística.

$$\begin{aligned}
\pi &= (0\ 2\ 3\ 7\ 6\ 4\ 5\ 1\ 8)\\
\pi \circ \tau(1,4,5) &= (0\ \underline{6}\ \underline{2\ 3\ 7}\ 4\ 5\ 1\ 8)\\
\pi \circ \tau(4,5,7) &= (0\ 2\ 3\ 7\ \underline{4\ 5}\ \underline{6}\ 1\ 8)\\
\pi \circ \tau(1,3,6) &= (0\ \underline{7\ 6\ 4}\ \underline{2\ 3}\ 5\ 1\ 7\ 8)
\end{aligned}$$

Quando queremos ordenar uma permutação usando o número mínimo de transposições, temos o problema de Ordenação de Permutações por Transposição. Quando nos restringimos às transposições determinísticas, temos os problemas de Ordenação de Permutações por Transposição Parcialmente Determinística e Ordenação de Permutações por Transposição Totalmente Determinística.

Precisamos definir mais alguns conceitos para lidar com transposições. Dada uma permutação π, um par de elementos consecutivos (π_i, π_{i+1}) é um *breakpoint de transposição* se $\pi_{i+1} - \pi_i \neq 1$. Denotamos o número de breakpoints de transposição de uma permutação π por $b_\tau(\pi)$. Note que a permutação identidade ι_n é a única permutação de tamanho n para a qual b_τ é 0. Uma sequência maximal de elementos de π sem breakpoints de transposição é chamada de uma *strip de transposição* de π.

Exemplo 12. Os breakpoints de transposição de uma permutação π. Cada breakpoint é indicado por um ponto (•). As strips de transposição de π são (0), (5), (1 2 3), (4) e (6 7).

$$\pi = (0\ \bullet\ 5\ \bullet\ 1\ 2\ 3\ \bullet\ 4\ \bullet\ 6\ 7)$$

5.1 Transposições Parcialmente Determinísticas

A seguir, propomos um algoritmo para ordenar permutações por transposições parcialmente determinísticas. O algoritmo se baseia no seguinte lema.

Lema 5 (Bafna-Pevzner [2])**.** *Uma transposição $\tau(i,j,k)$ aplicada em uma permutação π pode remover no máximo* 3 *breakpoints de transposição.*

Como consequência do Lema 5 temos que pelo menos $\frac{b_\tau(\pi)}{3}$ transposições são necessárias para ordenar uma permutação π. De fato, podemos garantir que a cada transposição pelo menos um breakpoint é removido, pois sempre podemos aplicar uma transposição que posiciona corretamente o primeiro elemento fora de ordem em π. Note que uma transposição $\tau(i,j,k)$ que posiciona corretamente o primeiro elemento fora de ordem π_i de π é necessariamente determinística, pois $\pi_j = i$.

O Algoritmo 4 apresenta um pseudocódigo do algoritmo desenvolvido. Como no caso de reversões determinísticas, a ideia é buscar por operações que removam o maior número de breakpoints. No pior caso uma transposição que remove 1 breakpoint será aplicada para posicionar corretamente o primeiro elemento fora de ordem em π.

Algoritmo 4: Ordenação por transposições parcialmente determinísticas

Entrada: Permutação π
Saída: Sequência de transposições determinísticas capazes de ordenar π

início
  S $\leftarrow$ sequência vazia
  enquanto π *não estiver ordenada* **faça**
    se *existe uma transposição determinística* $\tau(i,j,k)$ *que remove* 3 *breakpoints* **então**
      Adicione $\tau(i,j,k)$ em S
      $\pi \leftarrow \pi \circ \tau(i,j,k)$
    senão se *existe uma transposição determinística* $\tau(i,j,k)$ *que remove* 2 *breakpoint* **então**
      Adicione $\tau(i,j,k)$ em S
      $\pi \leftarrow \pi \circ \tau(i,j,k)$
    senão
      Seja π_i o primeiro elemento fora de ordem em π e j, tal que $\pi[j] = i$, e seja k o indice do primeiro breakpoint depois de $j+1$
      Adicione $\tau(i,j,k)$ em S
      $\pi \leftarrow \pi \circ \tau(i,j,k)$
  retorna S

Teorema 6. *Dada uma permutação π, o Algoritmo 4 para e encontra uma sequência que ordena π por transposições parcialmente determinísticas com tamanho no máximo $b_\tau(\pi)$.*

Demonstração. A cada passo o Algoritmo 4 aumenta em pelo menos 1 o número de breakpoints. Logo, como a permutação ι_n é a única com $b_\tau(\iota_n) = 0$, em no máximo $b_\tau(\pi)$ passos o algoritmo ordena a permutação. □

Corolário 1. *O Algoritmo 4 garante um fator de aproximação* 3 *para o problema de Ordenação por Transposição Determinística.*

Demonstração. Como consequência do Lema 5, $\frac{b_\tau(\pi)}{3}$ transposições são necessárias para ordenar π e pelo Teorema 6 o Algoritmo 4 consegue ordenar π com no máximo $b_\tau(\pi)$ transposições determinísticas. Portanto, o tamanho da sequência de transposições determinísticas obtidas pelo Algoritmo 4 está a um fator no máximo 3 do tamanho da menor sequência de transposições que ordena π. Como a menor sequência de transposições determinísticas que ordena π deve ser maior que a menor sequência de transposições que ordena π o resultado segue. □

Exemplo 13. A permutação $\pi = (0\ 3\ 2\ 1\ 6\ 4\ 5\ 7)$ é ordenada pelo algoritmo 4 com a

seguinte sequência de transposições parcialmente determinísticas.

$$\begin{aligned}
\pi &= (0\ 3\ 2\ 1\ 6\ 4\ 5\ 7)\\
\pi' = \pi \circ \tau(4,5,7) &= (0\ 3\ 2\ 1\ \underline{4\ 5}\ \underline{6}\ 7)\\
\pi'' = \pi' \circ \tau(1,3,4) &= (0\ \underline{1}\ \underline{3\ 2}\ 4\ 5\ 6\ 7)\\
\iota_n = \pi'' \circ \tau(2,3,4) &= (0\ 1\ \underline{2}\ \underline{3}\ 4\ 5\ 6\ 7)
\end{aligned}$$

Vamos analisar brevemente a complexidade de tempo do algoritmo 4. Note que em cada interação do laço das linhas 3 até 13, reduzimos em pelo menos 1 o número de breakpoints, portanto o laço é executado $O(n)$ vezes. Adicionalmente, em cada interação, como consideramos operações determinísticas, só temos que verificar as 2 transposições que posicionam corretamente cada elemento (uma considerando que o elemento é o primeiro da primeira sequência e outra considerando que o elemento é o primeiro da segunda sequência). Além disso, podemos verificar o número de breakpoits removidos por cada operação em tempo $O(1)$ se mantivermos um vetor representando a permutação π e um vetor representando a permutação inversa π^{-1}. Portanto, em cada iteração, encontrar a operação a ser aplicada leva tempo $O(n)$. Além disso, aplicar a operação encontrada para atualizar a nossa representação também leva tempo $O(n)$. No total a complexidade o algoritmo é $O(n^2)$.

5.2 Transposições Totalmente Determinísticas

Vamos descrever quais permutações podem ser ordenadas por transposições totalmente determinísticas utilizando as seguintes definições. Uma strip de transposição de uma permutação é uma *strip fixa* se ela tiver a forma $(k\ k+1\ \ldots\ \ell-1\ \ell)$, onde k e ℓ são a posição inicial e final da strip, respectivamente. Um par de strips de transposição consecutivas em uma permutação são um *par de strips trocadas* se eles têm a forma $(k\ k+1\ \ldots)$ e $(\ell\ \ell+1\ \ldots)$, onde ℓ é a posição inicial da primeira strip e $k = \ell + r$, sendo r o tamanho da segunda strip.

Teorema 7. *Uma permutação π pode ser ordenada por transposições totalmente determinísticas somente se toda strip for uma strip fixa ou pertencer a um par de strips trocadas.*

Demonstração. Por definição, uma transposição totalmente determinística $\tau(i,j,k)$ só pode ser aplicada em π se a sequência entre as posições i e $k-1$ é da forma $(j\ j+1\ \ldots\ k-1\ i\ i+1\ \ldots\ j-1)$. Note que, essa sequência deve estar contida em um par de strips trocadas para que ela esteja na posição esperada. Com esse fato, vamos realizar a prova por indução no número m de pares de strips trocadas de π.

> Base: se π não tem nenhum par de strips trocadas nenhuma transposição totalmente determinística pode ser aplicada e a permutação já deve estar ordenada, ou seja, tem apenas uma strip $(0\ \ldots\ n+1)$ que é fixa.
>
> Hipótese de Indução: toda a permutação π com $m-1, m > 0$ pares de strips trocadas e ordenável com a operação de transposição totalmente determinística é tal que toda strip é fixa ou pertence a um par de strips trocadas.

Passo: dada uma permutação π com $m > 0$ pares de strips trocadas. Seja $(\pi_i \ \ldots \ \pi_{j-1})$ e $(\pi_j \ \ldots \ \pi_{k-1})$ um par de strips trocadas de π e considere a permutação $\pi' = \pi \circ \tau(i, j, k)$ que terá $m - 1$ pares de strips trocadas. Pela hipótese de indução toda strip de π' é fixa ou pertence a um par de strips trocadas. Como toda strip de π diferente de $(\pi_i \ \ldots \ \pi_{j-1})$ e de $(\pi_j \ \ldots \ \pi_{k-1})$ também é strip de π', elas são fixas ou pertencem a pares de strips trocadas. □

Note que, dada uma permutação π no formato especificado pelo Teorema 7, podemos ordená-la por transposições totalmente determinísticas ao aplicar uma transposição em cada par de strips trocadas.

Exemplo 14. A permutação $\pi = (0\ 2\ 1\ 3\ 4\ 6\ 7\ 5\ 8)$ é ordenável pela seguinte sequência de transposições totalmente determinísticas.

$$\begin{aligned} \pi &= (0\ 2\ 1\ 3\ 4\ 6\ 7\ 5\ 8) \\ \pi' = \pi \circ \tau(1,2,3) &= (0\ \underline{1}\ \underline{2}\ 3\ 4\ 6\ 7\ 5\ 8) \\ \iota_n = \pi' \circ \tau(5,7) &= (0\ 1\ 2\ 3\ 4\ \underline{5}\ \underline{6\ 7}\ 8) \end{aligned}$$

Descrevemos agora o número T_n de permutações de tamanho n ordenáveis por reversões totalmente determinísticas. Temos os casos particulares $T_1 = 1$ e $T_2 = 2$, onde todas as permutações são ordenáveis. Para $n > 3$, apresentamos a seguir uma relação de recorrência. Consideramos $T_0 = 1$ e separamos as permutações em 3 casos:

- Existe uma permutação identidade ι_n.
- Existem $\sum_{k=2}^{n}(k-1)T_{n-k}$ permutações que começam com um par de strips trocadas. Para cada tamanho k do par, usamos a recorrência para contar as possibilidades para o restante da permutação e multiplicamos por $k-1$, para levar em conta a posição do breakpoint entre as duas strips. Note que um par de strips trocadas deve ter tamanho pelo menos 2.
- Existem $\sum_{k=2}^{n}\left(\sum_{i=1}^{k-2}(k-i-1)\right)T_{n-k}$ permutações que começam com uma strip fixa seguida por um par de strips trocadas. Para cada tamanho k da sequência strip e par, usamos a recorrência para contar as possibilidades para o restante da permutação e multiplicamos por $\sum_{i=1}^{k-2}(k-i-1)$, para levar em conta a posição dos breakpoints entre às três strips.

Juntando todos os casos temos a seguinte relação de recorrência:

$$T_n = 1 + \sum_{k=2}^{n}(k-1)T_{n-k} + \sum_{k=2}^{n}\left(\sum_{i=1}^{k-2}(k-i-1)\right)T_{n-k}$$

6 Testes Computacionais

Nesta seção apresentamos resultados dos testes experimentais para os problemas envolvendo reversão e transposição. Para esses testes desenvolvemos duas bases de dados e comparamos

os resultados dos nossos algoritmos com algoritmos que não tem a restrição de usar operações determinísticas.

6.1 Bases de Dados

Nesta seção, descrevemos brevemente as bases de dados desenvolvidas. Criamos a base REV para testar os algoritmos envolvendo reversões e a base TRANS para testar os algoritmos envolvendo transposições. Cada base de dados possui 40 conjuntos de permutações, sendo que cada conjunto foi construído com o uso de o operações e contém permutações de tamanho n. Para construir cada permutação, partimos da permutação inicial ι_n e aplicamos o operações. No caso da base REV são aplicadas o reversões, onde os índices i e j são escolhidos de forma aleatória e uniforme. No caso da base TRANS são aplicadas o transposições, onde os índices i, j e k são escolhidos de forma aleatória e uniforme.

6.2 Resultados Obtidos

Nós comparamos os resultados dos nossos algoritmos considerando operações determinísticas com variações desses algoritmos sem essa restrição. No caso de reversão, o algoritmo sem restrição busca por reversões quaisquer que removam dois breakpoints e, caso não existam reversões desse tipo, são aplicadas reversões que removem em média um breakpoint segundo a estratégia proposta por Kececioglu e Sankoff [11]. No caso de transposição, utilizamos o mesmo algoritmo do caso com operações determinísticas, mas buscamos por transposições quaisquer.

As tabelas 1 e 2 apresentam os resultados com reversão e transposição, respectivamente. Cada linha corresponde a um conjunto de pares de permutações e as colunas OP e $|\pi|$ indicam, respectivamente, o número de operações aplicadas para gerar as permutações do conjunto e o tamanho das permutações geradas. As próximas três colunas informam o mínimo (Min.), a média (Med.) e o máximo (Max.) das distâncias obtidas sem a restrição de operações determinísticas e considerando todos os pares do conjunto. As três colunas seguintes mostram os mesmos valores considerando operações determinísticas. As últimas seis colunas apresentam informações sobre a aproximação experimental das distâncias encontradas. A aproximação experimental é dada pela distância dividida pelo limitante inferior ($\frac{b_\rho(\pi)}{2}$ no caso de reversão e $\frac{b_\tau(\pi)}{3}$ no caso de transposição).

Percebemos que ao forçarmos o uso de operações determinísticas, o algoritmo vai tender a usar mais operações, o que era esperado. No entanto, forçar as operações determinísticas tem suas vantagens pois em geral os algoritmos ficam mais simples e eficientes, já que as escolhas são reduzidas substancialmente. Percebemos também que esse aumento na distância devido ao determinismo tende a ser maior no caso da reversão. Em média, a razão entre a média da transposição normal e da determinística é de 1.2, enquanto, no caso da reversão, é de 1.6.

7 Conclusão

Neste trabalho, exploramos as operações de troca, reversão e transposição, particularmente, nas suas versões determinísticas. Provamos que o Selection Sort faz apenas trocas parcialmente determinísticas, e ordena a permutação com o número mínimo dessas trocas.

Sendo a transposição e a reversão operações mais complexas, com as restrições de determinismo parcial e total focamos em gerar heurísticas e comparamos com heurísticas da literatura que estimam o valor mínimo de reversões ou transposições sem as restrições de determinismo.

Trabalhos futuros poderiam combinar as operações de troca e inserção através de uma heurística ponderada, assim como combinar as operações de transposição e reversão, sempre levando em conta que se podemos usar mais de um tipo de operação, o resultado deverá sempre ser igual ou melhor que as outras duas operações sozinhas, já que optar sempre só por um ou por outro é uma opção, constitui-se um limite superior a ser considerado.

Referências

[1] Martin Aigner e Douglas B West. "Sorting by insertion of leading elements". Em: *Journal of Combinatorial Theory, Series A* 45.2 (1987), pp. 306–309.

[2] Vineet Bafna e Pavel A. Pevzner. "Sorting by Transpositions". Em: *SIAM Journal on Discrete Mathematics* 11.2 (1998), pp. 224–240.

[3] David Berman, M. S. Klamkin, D. E. Knuth e J. H. Conway. "Problem 76-17". Em: *SIAM Review* 19.4 (1977), pp. 739–741.

[4] Piotr Berman, Sridhar Hannenhalli e Marek Karpinski. "1.375-Approximation Algorithm for Sorting by Reversals". Em: *Proceedings of the 10th Annual European Symposium on Algorithms (ESA'2002)*. London, UK: Springer-Verlag, 2002, pp. 200–210.

[5] Laurent Bulteau, Guillaume Fertin e Irena Rusu. "Sorting by Transpositions Is Difficult". Em: *SIAM Journal on Discrete Mathematics* 26.3 (2012), pp. 1148–1180.

[6] Alberto Caprara. "Sorting Permutations by Reversals and Eulerian Cycle Decompositions". Em: *SIAM Journal on Discrete Mathematics* 12.1 (1999), pp. 91–110.

[7] Isaac Elias e Tzvika Hartman. "A 1.375-Approximation Algorithm for Sorting by Transpositions". Em: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 3.4 (2006), pp. 369–379.

[8] Guillaume Fertin, Anthony Labarre, Irena Rusu, Eric Tannier e Stéphane Vialette. *Combinatorics of Genome Rearrangements*. 1ª ed. Computational Molecular Biology. London, England: The MIT Press, 2009.

[9] Philippe Flajolet e Robert Sedgewick. *Analytic Combinatorics*. 1ª ed. Cambridge University Press, 2009, p. 122.

[10] William H. Gates e Christos H. Papadimitriou. "Bounds for sorting by prefix reversal". Em: *Discrete Mathematics* 27.1 (1979), pp. 47–57.

[11] John Kececioglu e David Sankoff. “Exact and approximation algorithms for sorting by reversals, with application to genome rearrangement”. Em: *Algorithmica* 13.1-2 (1995), pp. 180–210.

[12] Donald E. Knuth. *The Art of Computer Programming*. Vol. 3. 1998.

[13] Geoffrey A. Watterson, Warren J. Ewens, Thomas E. Hall e Alexander Morgan. “The chromosome inversion problem”. Em: *Journal of Theoretical Biology* 99.1 (1982), pp. 1–7.

Tabela 1: Tamanhos das sequências de ordenação por reversões e reversões determinísticas para permutações da base REV.

OP	$\|\pi\|$	Distância						Aproximação Experimental					
		Reversão			Reversão Det.			Reversão			Reversão Det.		
		Min.	Avg.	Max.	Min.	Avg.	Max.	Min.	Avg.	Max.	Min.	Avg.	Max.
25	**50**	16	23.24	29	17	26.23	46	1.24	1.54	1.81	1.24	1.74	3.29
25	**100**	21	25.64	31	22	33.75	69	1.05	1.34	1.74	1.19	1.76	4.06
25	**150**	22	25.66	31	24	38.24	85	1.04	1.23	1.57	1.15	1.83	4.15
25	**200**	22	25.50	29	23	42.63	126	1.02	1.17	1.47	1.02	1.95	6.15
25	**250**	22	25.49	29	25	45.84	162	1.00	1.14	1.35	1.04	2.05	7.20
25	**300**	21	25.47	29	24	50.28	181	1.00	1.12	1.37	1.02	2.21	7.39
25	**350**	23	25.44	29	23	55.98	225	1.00	1.10	1.33	1.04	2.43	9.57
25	**400**	23	25.49	30	25	57.04	201	1.00	1.09	1.35	1.04	2.45	8.40
25	**450**	23	25.44	29	25	63.41	265	1.00	1.08	1.29	1.00	2.70	11.04
25	**500**	23	25.46	30	25	66.08	313	1.00	1.08	1.29	1.02	2.80	13.04
50	**50**	26	33.32	39	27	36.91	58	1.43	1.64	1.85	1.48	1.81	2.90
50	**100**	39	49.16	58	41	58.27	92	1.37	1.60	1.85	1.50	1.90	2.90
50	**150**	42	51.88	58	47	67.89	127	1.24	1.46	1.77	1.41	1.91	3.85
50	**200**	45	51.73	58	51	74.64	159	1.14	1.34	1.59	1.31	1.94	4.18
50	**250**	46	51.42	57	53	81.76	160	1.10	1.27	1.52	1.26	2.02	4.00
50	**300**	47	51.15	56	54	87.47	187	1.06	1.22	1.44	1.23	2.08	4.62
50	**350**	46	50.99	59	52	92.61	207	1.03	1.19	1.44	1.19	2.15	5.05
50	**400**	48	50.79	55	54	100.80	248	1.05	1.16	1.35	1.19	2.31	5.90
50	**450**	48	50.79	55	53	106.56	282	1.03	1.14	1.31	1.17	2.40	6.48
50	**500**	47	50.66	55	52	112.40	338	1.03	1.13	1.31	1.16	2.51	7.95
75	**50**	30	36.52	43	31	40.36	65	1.44	1.64	1.83	1.51	1.81	2.71
75	**100**	53	63.79	73	57	72.52	113	1.52	1.70	1.86	1.66	1.94	3.05
75	**150**	66	75.13	85	70	90.57	137	1.45	1.63	1.82	1.54	1.96	3.01
75	**200**	70	78.51	86	79	102.44	158	1.30	1.52	1.74	1.55	1.98	3.19
75	**250**	70	78.51	86	84	110.29	176	1.21	1.42	1.72	1.49	1.99	3.49
75	**300**	71	77.85	87	82	118.86	201	1.18	1.34	1.57	1.36	2.04	3.72
75	**350**	71	77.24	85	85	125.66	223	1.13	1.28	1.50	1.39	2.09	3.88
75	**400**	71	76.93	84	82	132.78	272	1.13	1.25	1.42	1.24	2.15	4.36
75	**450**	70	76.55	82	87	139.88	293	1.09	1.22	1.36	1.35	2.22	4.87
75	**500**	73	76.41	81	81	147.40	304	1.09	1.19	1.36	1.29	2.30	5.02
100	**50**	32	37.62	44	33	41.80	62	1.43	1.63	1.81	1.52	1.82	2.64
100	**100**	61	72.24	81	66	80.94	120	1.57	1.73	1.87	1.70	1.94	2.86
100	**150**	81	92.10	105	90	107.15	159	1.56	1.72	1.88	1.71	2.00	3.03
100	**200**	91	101.86	112	102	124.81	183	1.46	1.65	1.84	1.72	2.02	2.82
100	**250**	94	104.96	116	111	136.37	216	1.40	1.55	1.73	1.64	2.02	3.30
100	**300**	96	105.43	115	115	146.33	224	1.30	1.47	1.65	1.60	2.04	3.07
100	**350**	97	104.62	113	117	154.94	271	1.19	1.40	1.61	1.50	2.07	3.69
100	**400**	96	103.80	114	113	164.74	269	1.20	1.34	1.55	1.40	2.12	3.43
100	**450**	97	103.10	112	120	172.57	318	1.16	1.29	1.50	1.45	2.16	4.05
100	**500**	97	102.71	111	119	179.08	330	1.12	1.26	1.44	1.42	2.20	4.15

Tabela 2: Tamanhos das sequências de ordenação por transposições e transposições determinísticas para permutações da base TRANS.

OP	$\|\pi\|$	Distância						Aproximação Experimental					
		Transposição			Transposição Det.			Transposição			Transposição Det.		
		Min.	Avg.	Max.	Min.	Avg.	Max.	Min.	Avg.	Max.	Min.	Avg.	Max.
25	**50**	13	19.12	28	13	19.54	26	1.29	1.61	2.18	1.30	1.65	2.14
25	**100**	18	23.79	33	19	26.31	35	1.22	1.46	1.89	1.26	1.61	2.05
25	**150**	20	24.89	32	20	29.04	39	1.11	1.34	1.75	1.21	1.56	2.09
25	**200**	21	25.09	33	22	30.39	44	1.09	1.27	1.62	1.17	1.54	2.00
25	**250**	21	25.08	33	23	30.97	42	1.04	1.22	1.55	1.06	1.50	1.98
25	**300**	22	25.09	32	22	31.55	44	1.03	1.18	1.55	1.10	1.48	2.07
25	**350**	22	25.12	32	24	31.78	43	1.03	1.16	1.48	1.10	1.47	1.97
25	**400**	22	25.11	33	24	31.98	44	1.03	1.14	1.48	1.09	1.45	1.98
25	**450**	23	25.16	34	24	32.11	46	1.00	1.13	1.50	1.08	1.44	2.03
25	**500**	23	25.11	33	24	32.22	47	1.01	1.11	1.48	1.04	1.43	2.04
50	**50**	18	25.71	40	19	25.04	32	1.40	1.74	2.55	1.40	1.69	2.12
50	**100**	30	39.60	53	32	41.38	53	1.35	1.63	2.17	1.45	1.71	1.99
50	**150**	38	45.96	61	41	50.80	61	1.34	1.55	1.84	1.48	1.71	2.00
50	**200**	41	49.12	60	44	56.23	70	1.29	1.48	1.76	1.42	1.69	1.98
50	**250**	42	50.31	60	48	60.08	73	1.22	1.41	1.67	1.42	1.68	2.03
50	**300**	45	50.82	59	51	62.63	76	1.19	1.36	1.63	1.35	1.67	1.91
50	**350**	46	51.05	59	52	64.72	82	1.14	1.31	1.53	1.29	1.66	1.96
50	**400**	47	51.01	60	52	66.12	85	1.11	1.28	1.53	1.33	1.66	2.01
50	**450**	46	50.86	60	55	67.21	83	1.10	1.25	1.58	1.32	1.64	1.93
50	**500**	47	50.85	60	53	68.33	84	1.09	1.22	1.47	1.28	1.64	1.98
75	**50**	21	27.67	42	21	26.51	33	1.40	1.77	2.72	1.40	1.70	2.02
75	**100**	39	48.70	61	39	48.74	59	1.50	1.72	2.16	1.48	1.72	2.06
75	**150**	47	60.07	73	53	63.27	74	1.46	1.64	1.94	1.54	1.73	1.98
75	**200**	56	67.49	80	63	73.78	88	1.42	1.58	1.82	1.53	1.73	1.98
75	**250**	63	71.39	85	69	81.08	98	1.37	1.52	1.76	1.54	1.72	1.96
75	**300**	65	74.18	86	74	86.41	100	1.33	1.47	1.72	1.49	1.72	1.94
75	**350**	67	75.40	87	80	90.58	111	1.26	1.43	1.63	1.50	1.71	1.95
75	**400**	69	76.29	85	80	93.93	110	1.24	1.39	1.57	1.47	1.71	1.92
75	**450**	70	76.66	86	82	96.33	112	1.20	1.35	1.53	1.47	1.70	1.95
75	**500**	71	76.66	86	84	98.57	114	1.16	1.32	1.50	1.44	1.70	1.93
100	**50**	21	28.52	41	22	27.25	34	1.37	1.79	2.56	1.43	1.72	2.09
100	**100**	44	53.22	69	45	52.34	60	1.52	1.75	2.28	1.53	1.73	1.98
100	**150**	57	69.90	84	60	71.42	84	1.45	1.69	2.02	1.57	1.73	1.94
100	**200**	69	81.05	97	73	85.70	100	1.48	1.64	1.91	1.58	1.73	1.93
100	**250**	76	88.67	103	84	96.33	112	1.45	1.59	1.82	1.57	1.73	1.91
100	**300**	83	93.43	106	92	104.54	124	1.41	1.55	1.73	1.58	1.73	1.93
100	**350**	87	97.06	112	98	111.20	127	1.39	1.51	1.71	1.55	1.73	1.96
100	**400**	89	99.01	110	100	116.20	132	1.36	1.47	1.63	1.55	1.72	1.97
100	**450**	91	100.61	113	102	120.62	139	1.30	1.43	1.63	1.51	1.72	1.90
100	**500**	93	101.55	113	108	124.64	144	1.28	1.40	1.57	1.54	1.72	1.93