Problemas de Ordenação de Permutações por Operações Ponderadas

Alexsandro Oliveira Alexandrino

21 de Fevereiro de 2019

Orientador: Prof. Zanoni Dias
Coorientadora: Profa. Carla Negri Lintzmayer

Instituto de Computação - Universidade Estadual de Campinas

Roteiro

- Introdução
- Conceitos
- Operações Ponderadas pelo Número de Fragmentações
- Operações Curtas Ponderadas pelo Tamanho
- Conclusões
- Referências

Introdução

Introdução

- Distância Evolucionária e Distância de Rearranjo.
- Rearranjo de Genoma: mutação em larga escala que altera segmentos de um genoma.
    - Reversão.
    - Transposição.
- Representação de um genoma como uma permutação.
- Distância de ordenação de permutações.

Introdução

- Abordagem tradicional: a distância entre dois genomas é igual ao tamanho da sequência mínima de operações que transformam um genoma em outro.
  - Ordenação por Reversões: NP-Difícil [8];
  - Ordenação por Reversões com Sinais: P [13];
  - Ordenação por Transposições: NP-Difícil [7];
  - Ordenação por Reversões e Transposições: Complexidade em aberto.
- Abordagens ponderadas: a distância entre dois genomas é igual ao custo de uma sequência de operações de custo mínimo que transforma um genoma em outro.

Introdução

- As funções de custo mais estudas são:
  - Relacionadas ao tamanho do rearranjo [4, 19];
  - Relacionadas ao tipo do rearranjo [2, 11].

Introdução

- Nesse trabalho, consideramos duas variações do problema:
    - Ordenação de Permutações por Operações Ponderadas pelo Número de Fragmentações;
    - Ordenação de Permutações por Operações Curtas Ponderadas pelo Tamanho.

Conceitos

Representação de um Genoma - Permutações com Sinais

- $\pi = (\pi_1\ \pi_2\ \pi_3\ \ldots\ \pi_n)$
- $\pi_i \in \{-n,\ -(n-1),\ \ldots,\ -1,\ +1,\ \ldots,\ +(n-1),\ +n\}$
- $|\pi_i| = |\pi_j| \leftrightarrow i = j$

Permutações com Sinais

- $\pi = (+5\ +2\ +4\ -1\ -3)$

Permutações

Representação de um Genoma - Permutações sem Sinais

- $\pi = (\pi_1\ \pi_2\ \pi_3\ \ \ldots\ \ \pi_n)$
- $\pi_i \in \{1,\ \ldots,\ n-1,\ n\}$
- $\pi_i = \pi_j \leftrightarrow i = j$

Permutações sem Sinais

- $\pi = (5\ 2\ 4\ 1\ 3)$

Permutações

Permutação identidade

- $\iota = (1\ 2\ 3\ \cdots\ n)$

Permutação reversa

- $\eta = (n\ \ldots\ 2\ 1)$
- $\overline{\eta} = (-n\ \ldots\ -2\ -1)$

Modelo de Rearranjo

- Conjunto de rearranjos permitidos para o cálculo da distância;
- Exemplos de modelos de rearranjo: r, $\bar{r}$, t, rt e $\bar{r}t$.
- Ao considerar apenas operações curtas, adicionamos à notação do modelo o prefixo s (curtas, do inglês *short*).

Reversão

Definições

- Uma reversão sem sinais $\rho(i,j)$, com $1 \leq i < j \leq n$, inverte a ordem do segmento $(\pi_i, \ldots, \pi_j)$.

$$\pi = (\pi_1 \;\ldots\; \pi_{i-1} \; \underline{\pi_i \;\ldots\; \pi_j} \; \pi_{j+1} \;\ldots\; \pi_n)$$

$$\pi \circ \rho(i,j) = (\pi_1 \;\ldots\; \pi_{i-1} \; \underline{\pi_j \;\ldots\; \pi_i} \; \pi_{j+1} \;\ldots\; \pi_n)$$

- Reversão de prefixo: $\rho(1,j)$, com $1 < j \leq n$.
- Reversão de sufixo: $\rho(i,n)$, com $1 \leq i < n$.

Exemplo

$$\pi = (5\ 4\ \underline{2\ 3}\ 1)$$

$$\pi \circ \rho(3,4) = (5\ 4\ 3\ 2\ 1)$$

Reversão

Definições

- Uma reversão com sinais $\bar{\rho}(i,j)$, com $1 \leq i \leq j \leq n$, inverte a ordem do segmento $(\pi_i, \ \ldots, \pi_j)$ e troca o sinal dos elementos desse segmento.

$$\pi = (+\pi_1 \ \ldots \ +\pi_{i-1} \ \underline{+\pi_i \ \ldots \ +\pi_j} \ +\pi_{j+1} \ \ldots \ +\pi_n)$$

$$\pi \circ \bar{\rho}(i,j) = (+\pi_1 \ \ldots \ +\pi_{i-1} \ \underline{-\pi_j \ \ldots \ -\pi_i} \ +\pi_{j+1} \ \ldots \ +\pi_n)$$

- Reversão com sinais de prefixo: $\bar{\rho}(1,j)$, com $1 \leq j \leq n$.
- Reversão com sinais de sufixo: $\bar{\rho}(i,n)$, com $1 \leq i \leq n$.

Exemplo

$$\pi = (-5 \ -4 \ \underline{+2 \ +3} \ -1)$$

$$\pi \circ \bar{\rho}(3,4) = (-5 \ -4 \ -3 \ -2 \ -1)$$

Transposição

Definições

- Uma transposição $\tau(i,j,k)$, com $1 \leq i < j < k \leq n+1$, troca de posição o segmento $(\pi_i, \pi_{i+1}, \ldots, \pi_{j-1})$ com o segmento adjacente $(\pi_j, \pi_{j+1}, \ldots, \pi_{k-1})$.

$$\pi = (\pi_1 \;\ldots\; \pi_{i-1}\; \underline{\pi_i \;\ldots\; \pi_{j-1}}\; \underline{\pi_j \;\ldots\; \pi_{k-1}}\; \pi_k \;\ldots\; \pi_n)$$

$$\pi \circ \tau(i,j,k) = (\pi_1 \;\ldots\; \pi_{i-1}\; \underline{\pi_j \;\ldots\; \pi_{k-1}}\; \underline{\pi_i \;\ldots\; \pi_{j-1}}\; \pi_k \;\ldots\; \pi_n)$$

- Transposição de prefixo: $\tau(1,j,k)$, com $1 < j < k \leq n+1$.
- Transposições de sufixo: $\tau(i,j,n+1)$, com $1 \leq i < j \leq n$.

Exemplo

$$\pi = (\underline{4\ 5}\ \underline{1\ 2\ 3})$$

$$\pi \circ \tau(1,3,6) = (1\ 2\ 3\ 4\ 5)$$

Rearranjos

Definições

- Uma operação é completa se envolve todos os elementos da permutação.
- O tamanho $|\beta|$ de uma operação β é igual a quantidade de elementos afetados por ela.
  - $|\rho(i,j)| = j - i + 1$.
  - $|\tau(i,j,k)| = k - i$.
- Uma operação β é super curta se $|\beta| \leq 2$ e curta se $|\beta| \leq 3$.

Função de Custo

- $f : M \to \mathbb{R}$ é a função que denota o custo associado a cada operação de M.
- O custo de uma sequência de operações $S = \langle \beta_1, \beta_2, \ldots, \beta_\ell \rangle$ é igual a $f(S) = \sum_{i=1}^{\ell} f(\beta_i)$.

Ordenação de Permutações

Distância de Ordenação

- Dados um modelo de rearranjo M, uma função de custo f e uma permutação π, a distância de ordenação $d_M^f(\pi)$ é o custo de uma sequência de operações $S = \langle \beta_1, \beta_2, \ldots, \beta_\ell \rangle$ pertencentes a M, tal que $\pi \circ S = \iota$ e $\sum_{i=1}^{\ell} f(\beta_i)$ é mínimo.

Diâmetro

- Dados um modelo de rearranjo M e uma função de custo f, o diâmetro de tamanho n é denotado por $D_M^f(n) = \max\{d_M^f(\pi) : \pi \text{ tem tamanho } n\}$, com $n \in \mathbb{Z}^+$. Ou seja, $D_M^f(n)$ é a maior distância $d_M^f(\pi)$ entre todas as permutações de tamanho n.

Operações Ponderadas pelo Número de Fragmentações

Operações Ponderadas pelo Número de Fragmentações

- Nova ponderação proporcional à quantidade de fragmentações que uma operação causa na permutação.
- Similar à ponderação por tipo de rearranjo [2, 6, 11].

Fragmentações

Para qualquer par de posições $(i, i+1)$ de uma permutação π, um rearranjo β causa fragmentação entre $(i, i+1)$ se π_i e π_{i+1} não são adjacentes em $\pi \circ \beta$.

Operações Ponderadas pelo Número de Fragmentações

$$\begin{aligned}
\pi &= (\pi_1 \ \dots \ \pi_{i-1} \ \pi_i \ \pi_{i+1} \ \dots \ \pi_{j-1} \ \pi_j \ \pi_{j+1} \ \dots \ \pi_n) \\
\pi \circ \rho(1, n) &= (\underline{\pi_n \ \dots \ \pi_{j+1} \ \pi_j \ \pi_{j-1} \ \dots \ \pi_{i+1} \ \pi_i \ \pi_{i-1} \ \dots \ \pi_1}) \\
\pi \circ \rho(1, j) &= (\underline{\pi_j \ \pi_{j-1} \ \dots \ \pi_{i+1} \ \pi_i \ \pi_{i-1} \ \dots \ \pi_1} \ \pi_{j+1} \ \dots \ \pi_n) \\
\pi \circ \rho(i, n) &= (\pi_1 \ \dots \ \pi_{i-1} \ \underline{\pi_n \ \dots \ \pi_{j+1} \ \pi_j \ \pi_{j-1} \ \dots \ \pi_{i+1} \ \pi_i}) \\
\pi \circ \rho(i, j) &= (\pi_1 \ \dots \ \pi_{i-1} \ \underline{\pi_j \ \pi_{j-1} \ \dots \ \pi_{i+1} \ \pi_i} \ \pi_{j+1} \ \dots \ \pi_n)
\end{aligned}$$

Reversões

O custo de uma reversão $\rho(i, j)$ é:

- $f(\rho(i, j)) = 0$, se $i = 1$ e $j = n$
- $f(\rho(i, j)) = 1$, se $i = 1$ e $j < n$
- $f(\rho(i, j)) = 1$, se $i > 1$ e $j = n$
- $f(\rho(i, j)) = 2$, se $i > 1$ e $j < n$

Operações Ponderadas pelo Número de Fragmentações

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

$$\pi \circ \tau(1, j, n+1) = (\underline{\pi_j \ \pi_{j+1} \ \dots \ \pi_{k-1} \ \pi_k \ \dots \ \pi_n} \ \underline{\pi_1 \ \dots \ \pi_{i-1} \ \pi_i \ \pi_{i+1} \ \dots \ \pi_{j-1}})$$

$$\pi \circ \tau(1, j, k) = (\underline{\pi_j \ \pi_{j+1} \ \dots \ \pi_{k-1}} \ \underline{\pi_1 \ \dots \ \pi_{i-1} \ \pi_i \ \pi_{i+1} \ \dots \ \pi_{j-1}} \ \pi_k \ \dots \ \pi_n)$$

$$\pi \circ \tau(i, j, n+1) = (\pi_1 \ \dots \ \pi_{i-1} \ \underline{\pi_j \ \pi_{j+1} \ \dots \ \pi_{k-1} \ \pi_k \ \dots \ \pi_n} \ \underline{\pi_i \ \pi_{i+1} \ \dots \ \pi_{j-1}})$$

$$\pi \circ \tau(i, j, k) = (\pi_1 \ \dots \ \pi_{i-1} \ \underline{\pi_j \ \pi_{j+1} \ \dots \ \pi_{k-1}} \ \underline{\pi_i \ \pi_{i+1} \ \dots \ \pi_{j-1}} \ \pi_k \ \dots \ \pi_n)$$

Transposições

O custo de uma transposição $\tau(i, j, k)$ é:

- $f(\tau(i, j, k)) = 1$, se $i = 1$ e $k = n + 1$
- $f(\tau(i, j, k)) = 2$, se $i = 1$ e $k < n + 1$
- $f(\tau(i, j, k)) = 2$, se $i > 1$ e $k = n + 1$
- $f(\tau(i, j, k)) = 3$, se $i > 1$ e $k < n + 1$

Operações Ponderadas pelo Número de Fragmentações

- O objetivo dos problemas da Ordenação por Operações Ponderadas pelo Número de Fragmentações é encontrar a distância $d_M^f(\pi)$, onde f é a função de custos associada ao número de fragmentações que uma operação causa na permutação.
- Apresentamos cinco variações do problema:
  - Reversões (SbFWR);
  - Reversões com Sinais (SbFW$\bar{\text{R}}$);
  - Transposições (SbFWT);
  - Reversões e Transposições (SbFWRT);
  - Reversões com Sinais e Transposições (SbFW$\bar{\text{R}}$T).

Relação com a Abordagem Tradicional

Lema 3.1

Considerando os problemas SbR e Sb$\bar{R}$, um algoritmo de α-aproximação para a versão não ponderada do problema é uma 2α-aproximação assintótica para a versão ponderada por fragmentações.

Lema 3.2

Considerando o problema SbT, um algoritmo de α-aproximação para a versão não ponderada do problema é uma 3α-aproximação para a versão ponderada por fragmentações.

Lema 3.3

Considerando os problemas SbRT e Sb$\bar{R}T$, um algoritmo de α-aproximação para a versão não ponderada do problema é uma 3α-aproximação assintótica para a versão ponderada por fragmentações.

Tabela 1: Fatores de aproximação dos problemas ponderados pelo número de fragmentações usando os melhores algoritmos conhecidos para a versão não ponderada.

	Melhor Algoritmo	
	Problema Não Ponderado	**Problema Ponderado**
SbFWR	1.375-aproximação [5]	2.75-aproximação assintótica (Lema 3.1)
SbFW$\bar{\text{R}}$	Algoritmo exato [13]	2-aproximação assintótica (Lema 3.1)
SbFWT	1.375-aproximação [10]	4.125-aproximação (Lema 3.2)
SbFWRT	$(2.8334 + \epsilon)$-aproximação, para $\epsilon > 0$ [9, 24]	$(8.5002 + \epsilon)$-aproximação assintótica, para $\epsilon \geq 0$ (Lema 3.3)
SbFW$\bar{\text{R}}$T	2-aproximação [26]	6-aproximação assintótica (Lema 3.3)

Limitantes Inferiores

Definição

- Um breakpoint de reversão sem sinais existe entre um par de elementos consecutivos π_i e π_{i+1} se $|\pi_{i+1} - \pi_i| \neq 1$, para $0 \leq i \leq n$.
- O número de breakpoints de reversão em π é denotado por $b_r(\pi)$.
- Um breakpoint de transposição (ou de reversão com sinais) existe entre um par de elementos consecutivos π_i e π_{i+1} se $\pi_{i+1} - \pi_i \neq 1$, para $0 \leq i \leq n$.
- O número de breakpoints de transposição (ou de reversão com sinais) em π é denotado por $b_t(\pi)$ (ou $b_{\bar{r}}(\pi)$).
- Uma strip é uma subsequência maximal de π sem breakpoints.

Exemplo

$$\pi \; = \; (-2\ -1 \cdot +4\ +5 \cdot -3)$$

$$b_{\bar{r}}(\pi) \; = \; 2, strips(\pi) \; = \; \{(-2\ -1), (+4\ +5), (-3)\}$$

Lema 3.4

Para qualquer permutação π e operação β, o custo médio para remoção de um breakpoint é maior ou igual a 1.

Lema 3.5

Para qualquer permutação π, $b_r(\pi) \leq d_r^f(\pi)$ e $b_{\bar{r}}(\pi) \leq d_{\bar{r}}^f(\pi)$.

Lema 3.6

Para qualquer permutação π, $b_t(\pi) \leq d_t^f(\pi)$.

Lema 3.7

Para qualquer permutação π, $b_r(\pi) \leq d_{rt}^f(\pi)$ e $b_{\bar{r}}(\pi) \leq d_{\bar{r}t}^f(\pi)$.

Algoritmos Básicos de 2-Aproximação

Visão Geral

Enquanto $b(\pi) > 0$:

1. Seja $s = (\pi_i, \ldots, \pi_j)$ a strip contendo o elemento com maior valor absoluto fora de posição;
2. Mova a strip s para a sua posição correta com custo de no máximo 2.

- Esses algoritmos são chamados de 2-R, 2-$\bar{\text{R}}$ e 2-T.
- O custo médio para remoção de um breakpoint é menor ou igual a 2.

Teorema 3.8

2-R, 2-$\bar{R}$ e 2-T são algoritmos de 2-aproximação para os seus respectivos problemas.

Algoritmos Gulosos de 2-Aproximação

- Escolha gulosa: aplicar operação com melhor razão “breakpoints removidos / custo da operação”.
- Repita até que não existam mais breakpoints na permutação.
- As permutações η e $\bar{\eta}$ são ordenadas com uma reversão de custo 0.
- Esses algoritmos são chamados 2-Rg, 2-$\bar{\text{R}}$g, 2-Tg, 2-RTg, e 2-$\bar{\text{R}}$Tg.

- Para os problemas SbFWT, SbFWRT e SbFW$\bar{\text{R}}$T, sempre existe operação de custo 2 que remove pelo menos um breakpoint.
- Para os problemas SbFWR e SbFW$\bar{\text{R}}$, os algoritmos podem chegar a um estado em que não existem reversões que possam remover breakpoints.
- Nesse caso, o algoritmo aplica uma reversão de custo 1 a fim de garantir que exista operação com razão 1 na próxima iteração.
- Para todos os algoritmos, o custo médio para remoção de um breakpoint é menor ou igual a 2.

Teorema 3.9

2-Rg, 2-$\bar{R}$g, 2-Tg, 2-RTg, e 2-$\bar{R}$Tg são algoritmos de 2-aproximação para os seus respectivos problemas.

Outros Resultados

- Resultados experimentais dos algoritmos desenvolvidos.
- Estudo da classe de permutações chamada de permutações simples:
  - O número de permutações simples sem sinais de tamanho n, com $n > 3$, é maior ou igual a $\lfloor \frac{n}{3} \rfloor !$;
  - O número de permutações simples com sinais de tamanho n, com $n > 3$, é maior ou igual a $\lfloor \frac{n}{3} \rfloor ! \times 2^{2\lfloor n/3 \rfloor}$;
  - Algoritmo de 1.5-aproximação assintótica para SbFWT e SbFW$\bar{\text{R}}$T, considerando apenas permutações simples;
  - Algoritmo de 9/4-aproximação assintótica para SbFWT e SbFW$\bar{\text{R}}$T, considerando qualquer permutação.

Outros Resultados

Tabela 2: Resumo dos limitantes para o diâmetro dos problemas SbFWR, SbFWT, SbFWRT, SbFW$\bar{\text{R}}$ e SbFW$\bar{\text{R}}$T.

Problema	**Limitante Inferior**	**Limitante Superior**
SbFWR	$n-1$	$\frac{18n}{11} + O(1)$
SbFWT	n	$2(n - \log_{7/2} n)$
SbFWRT	$n-1$	$\frac{18n}{11} + O(1)$
SbFW$\bar{\text{R}}$	n	$2n - 6$
SbFW$\bar{\text{R}}$T	n	$2n - 6$

Operações Curtas Ponderadas pelo Tamanho

Operações Curtas Ponderadas pelo Tamanho

- Indícios sugerem que, em alguns casos, as operações que ocorreram no processo evolutivo não agem em segmentos muito longos do genoma [6, 17].
- Essa observação motivou os problemas:
  - Ordenação de Permutações por Operações Ponderadas pelo Tamanho [12, 14, 15, 16, 25];
  - Ordenação de Permutações por Operações de Tamanho Limitado [20, 22, 23].
- Estudamos problemas que combinam a ponderação pelo tamanho do rearranjo com a restrição de que apenas operações curtas são permitidas:
  - Ordenação de Permutações por Operações Curtas Ponderadas pelo Tamanho.

Operações Curtas Ponderadas pelo Tamanho

- O objetivo desses problemas é encontrar a distância $d_M^{\ell}(\pi)$, onde ℓ é a ponderação por tamanho e M possui apenas operações curtas.
- Apresentamos cinco variações do problema:
  - Reversões Curtas (SbLWsR);
  - Reversões com Sinais Curtas (SbLWs$\bar{\text{R}}$);
  - Transposições Curtas (SbLWsT);
  - Reversões Curtas e Transposições Curtas (SbLWsRsT);
  - Reversões com Sinais Curtas e Transposições Curtas (SbLWs$\bar{\text{R}}$sT).

Relação com a Abordagem Não Ponderada

Lema 4.1

Considerando os problemas SbLWsR, SbLWsT e SbLWsRsT, uma α-aproximação para a versão não ponderada do problema é uma 1.5α-aproximação para a versão ponderada pelo tamanho.

Lema 4.2

Considerando os problemas SbLWs$\bar{R}$ e SbLWs$\bar{R}$sT, uma α-aproximação para a versão não ponderada do problema é uma 3α-aproximação para a versão ponderada pelo tamanho.

Relação com a Abordagem Não Ponderada

Tabela 3: Fatores de aproximação dos problemas com operações curtas ponderadas pelo tamanho usando os melhores algoritmos conhecidos para a versão não ponderada.

	Melhor Algoritmo	
	Problema Não Ponderado	**Problema Ponderado**
SbLWsR	2-aproximação [14]	3-aproximação (Lema 4.1)
SbLWs$\bar{\text{R}}$	5-aproximação [12]	15-aproximação (Lema 4.2)
SbLWsT	5/4-aproximação [16]	15/8-aproximação (Lema 4.1)
SbLWsRsT	2-aproximação [25]	3-aproximação (Lema 4.1)
SbLWs$\bar{\text{R}}$sT	3-aproximação [12]	9-aproximação (Lema 4.2)

Conceitos

Inversões

Um par de elementos (π_i, π_j) de uma permutação π é uma *inversão* se $i < j$ e $|\pi_i| > |\pi_j|$. Usamos $\mathrm{Inv}(\pi)$ para denotar o número de inversões em uma permutação π.

Exemplo

A permutação $\pi = (3\ 4\ 1\ 2)$ possui as inversões $(\pi_1 = 3, \pi_3 = 1)$, $(\pi_1 = 3, \pi_4 = 2)$, $(\pi_2 = 4, \pi_3 = 1)$ e $(\pi_2 = 4, \pi_4 = 2)$. Neste exemplo, $\mathrm{Inv}(\pi) = 4$.

Inversões

Dada uma permutação π, a variação no número de inversões causada por uma operação β é denotada por $\Delta\mathrm{Inv}(\pi, \beta) = \mathrm{Inv}(\pi) - \mathrm{Inv}(\pi \circ \beta)$.

Conceitos

Grafo de Inversões

O *grafo de inversões* de uma permutação π é o grafo não direcionado $G^{inv}(\pi) = (V, E)$, onde $V = \{\pi_1, \pi_2, \ldots, \pi_n\}$ e $E = \{(\pi_i, \pi_j) : (\pi_i, \pi_j)$ é uma inversão em $\pi\}$.

- Um componente conexo C de $G^{inv}(\pi)$ é *ímpar* se ele contém um número ímpar de elementos negativos (vértices negativos) e é *par* caso contrário.
- $c^{inv}(\pi)$ e $c^{inv}_{odd}(\pi)$ denotam o número de componentes conexos e o número de componentes conexos ímpares de $G^{inv}(\pi)$, respectivamente.
- O número de vértices não isolados de $G^{inv}(\pi)$ é denotado por $\gamma(\pi)$.

Conceitos

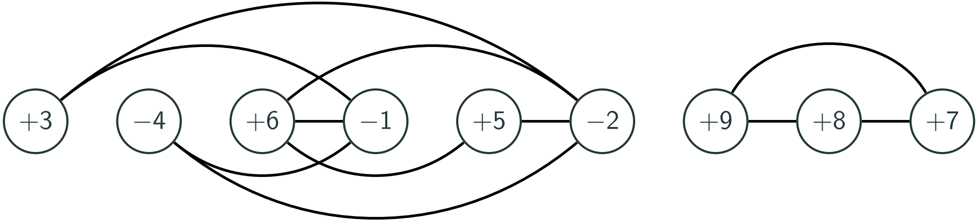

Figura 1: Grafo de inversões para $\pi = (+3\ -4\ +6\ -1\ +5\ -2\ +9\ +8\ +7)$. Neste exemplo, $c^{inv}(\pi) = 2$, $c^{inv}_{odd}(\pi) = 1$ e $\gamma(\pi) = 9$.

Conceitos

Entropia

A *entropia* do elemento π_i é definida como $\mathrm{ent}(\pi_i) = ||\pi_i| - i|$, para $1 \leq i \leq n$.

Definimos os conjuntos:

- $E_\pi^{even^-} = \{\pi_i : \pi_i < 0 \text{ e } \mathrm{ent}(\pi_i) \text{ é par}\}$;
- $E_\pi^{odd^+} = \{\pi_i : \pi_i > 0 \text{ e } \mathrm{ent}(\pi_i) \text{ é ímpar}\}$.

Exemplo

Para $\pi = (+5\ -4\ +3\ -1\ +2)$, temos $\mathrm{ent}(+5) = 4$, $\mathrm{ent}(-4) = 2$, $\mathrm{ent}(+3) = 0$, $\mathrm{ent}(-1) = 3$, e $\mathrm{ent}(+2) = 3$, e os conjuntos $E_\pi^{even^-} = \{-4\}$ e $E_\pi^{odd^+} = \{+2\}$.

Lema 4.3 (Galvão *et al.* [12])

Para qualquer permutação com sinais π e reversão super curta $\bar{\rho}$, temos que $|E_\pi^{even^-}| + |E_\pi^{odd^+}| = |E_{\pi'}^{even^-}| + |E_{\pi'}^{odd^+}|$, *onde* $\pi' = \pi \circ \bar{\rho}$.

Algoritmos para Permutações sem Sinais

Lema 4.4 (Galvão *et al.* [12])

Para qualquer permutação π *e operação curta* β,

i) $-1 \leq \Delta\mathrm{Inv}(\pi, \beta) \leq 1$ *se* β *tem tamanho* 2*;*

ii) $-2 \leq \Delta\mathrm{Inv}(\pi, \beta) \leq 2$ *se* β *é uma transposição de tamanho* 3*;*

iii) $-3 \leq \Delta\mathrm{Inv}(\pi, \beta) \leq 3$ *se* β *é uma reversão de tamanho* 3*.*

Lema 4.5

Para qualquer permutação π, *se* $\mathrm{Inv}(\pi) > 0$, *então existe uma inversão* (π_i, π_{i+1}).

Algoritmos para Permutações sem Sinais

- Escolha gulosa: aplicar operação com melhor razão $\frac{\Delta \mathrm{Inv}(\pi,\beta)}{|\beta|}$.
- Pelo Lema 4.5, sempre existe reversão ou transposição curta com $\frac{\Delta \mathrm{Inv}(\pi,\beta)}{|\beta|} \geq \frac{1}{2}$.
- A permutação identidade é a única permutação com $\mathrm{Inv}(\iota) = 0$.
- Repita até que a permutação esteja ordenada.
- Esses algoritmos são chamados 2-sR, $4/3$-sT e 2-sRsT.

Algoritmos para Permutações sem Sinais

Teorema 4.6

2-sR e 2-sRsT são algoritmos de 2-aproximação para SbLWsR e SbLWsRsT, respectivamente.

Teorema 4.7

4/3-sT é uma algoritmo de 4/3-aproximação para SbLWsT.

Função Objetivo

Dada uma permutação com sinais π e uma sequência S, seja $\pi' = \pi \circ S$. A função objetivo é definida como

$$\phi(\pi, S) = \frac{(2\,\mathrm{Inv}(\pi) + |E_{\pi}^{even^-}| + |E_{\pi}^{odd^+}|) - (2\,\mathrm{Inv}(\pi') + |E_{\pi'}^{even^-}| + |E_{\pi'}^{odd^+}|)}{\sum_{\beta \in S} |\beta|}.$$

Algoritmos para Permutações com Sinais

Lema 4.8

Para qualquer permutação com sinais π *e qualquer operação curta* β*, temos que* $\phi(\pi, \beta) \leq 3$.

Lema 4.9

Para qualquer permutação com sinais $\pi \neq \iota$*, existe reversão com sinais curta* $\bar{\rho}$ *com* $\phi(\pi, \bar{\rho}) \geq 1$.

Algoritmos para Permutações com Sinais

- A permutação identidade é a única permutação com $\mathrm{Inv}(\iota) = |E_\iota^{even^-}| = |E_\iota^{odd^+}| = 0$.
- Se S é uma sequência de ordenação ótima, então $\phi(\pi, S) \geq \phi(\pi, S')$, para qualquer sequência de ordenação S'.
- Escolha gulosa: aplicar a operação curta β com maior valor objetivo $\phi(\pi, \beta)$.
- Esses algoritmos são chamados 3-s$\bar{\text{R}}$ e 3-s$\bar{\text{R}}$sT.

Teorema 4.10

3-s$\bar{R}$ e 3-s$\bar{R}$sT são algoritmos de 3-aproximação para SbLWs$\bar{R}$ e SbLWs$\bar{R}$sT, respectivamente.

Algoritmos para Permutações com Sinais

Função Objetivo

Dada uma permutação com sinais π e uma sequência S, seja $\pi' = \pi \circ S$. A função objetivo é definida como

$$\psi(\pi, S) = \frac{(2\,\mathrm{Inv}(\pi) + c_{odd}^{inv}(\pi)) - (2\,\mathrm{Inv}(\pi') + c_{odd}^{inv}(\pi'))}{\sum_{\beta \in S} |\beta|}.$$

Lema 4.11

Para qualquer permutação com sinais π e qualquer operação curta β, temos que $\psi(\pi, \beta) \leq \frac{7}{3}$.

Lema 4.12

Para qualquer permutação com sinais $\pi \neq \iota$, existe operação curta β com $\psi(\pi, \beta) \geq 1$.

Algoritmos para Permutações com Sinais

- A permutação identidade é a única permutação com $\mathrm{Inv}(\iota) = c^{inv}_{odd}(\iota) = 0$.
- Se S é uma sequência de ordenação ótima, então $\psi(\pi, S) \geq \psi(\pi, S')$, para qualquer sequência de ordenação S'.
- Escolha gulosa: aplicar a operação curta β com maior valor objetivo $\psi(\pi, \beta)$.
- Esse algoritmo é chamado de 7/3-s$\bar{\text{R}}$.

Teorema 4.13

7/3-s$\bar{R}$sT é um algoritmo de 7/3-aproximação para SbLWs$\bar{R}$sT.

Outros Resultados

- $(1 + \gamma(\pi)/(3\,\mathrm{Inv}(\pi)))$-aproximação para SbLWsT.
- Resultados experimentais dos algoritmos desenvolvidos.
- Análise dos problemas quando a função de custo é igual a ℓ^α, onde ℓ é o tamanho da operação e $\alpha \geq 1$ é uma constante.

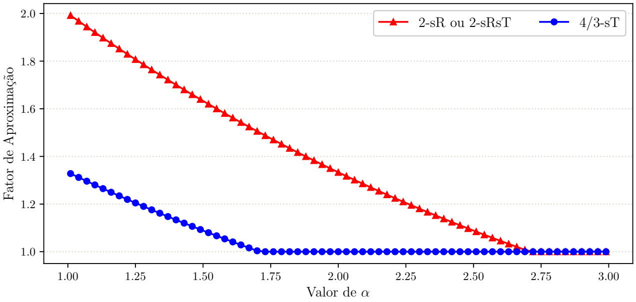

Figura 2: Fator de aproximação dos algoritmos para permutações sem sinais, para valores de $\alpha \geq 1$.

Outros Resultados

- Além disso, demonstramos que existe algoritmo polinomial para SbLWs$\bar{\text{R}}$ quando $\alpha \geq 3$.

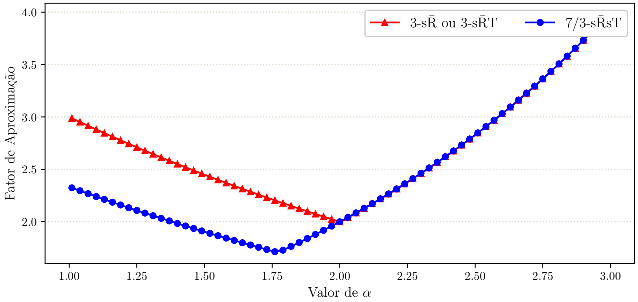

Figura 3: Fator de aproximação dos algoritmos para permutações com sinais, para valores de $\alpha \geq 1$.

Conclusões

Resumo dos Resultados

- Ordenação de Permutações por Operações Ponderadas pelo Número de Fragmentações:
  - Relação com a abordagem não ponderada;
  - Algoritmos de 2-aproximação para cinco modelos de rearranjo;
  - Resultados experimentais desses algoritmos;
  - Algoritmos de 1.5-aproximação para dois modelos de rearranjo, considerando permutações simples;
  - Limitantes para o diâmetro dos cinco problemas.
- Ordenação de Permutações por Operações Curtas Ponderadas pelo Tamanho:
  - Relação com a abordagem não ponderada;
  - Algoritmos de aproximação constante para cinco modelos de rearranjo;
  - Resultados experimentais desses algoritmos;
  - Análise dos problemas quando a função de custo é igual a ℓ^α, onde ℓ é o tamanho da operação e $\alpha \geq 1$ é uma constante.

Publicações

- *Approximation Algorithms for Sorting Permutations by Fragmentation-Weighted Operations* [1], apresentado na *International Conference on Algorithms for Computational Biology* (AlCoB'2018).
  - Versão estendida submetida para uma revista internacional.
- *Approximation Algorithms for Sorting Permutations by Length-Weighted Short Rearrangements*, aceito para apresentação no *Latin and American Algorithms, Graphs and Optimization Symposium* (LAGOS'2019).
- *Sorting λ-Permutations by λ-Operations* [21], apresentado no *Brazilian Symposium on Bioinformatics* (BSB'2018).
  - Colaboração com Guilherme Henrique Santos Miranda.
  - Versão estendida submetida para uma revista internacional.

Obrigado

Referências

[1] Alexsandro Oliveira Alexandrino, Carla Negri Lintzmayer, and Zanoni Dias. Approximation Algorithms for Sorting Permutations by Fragmentation-Weighted Operations. In *Algorithms for Computational Biology*, volume 10849, pages 53–64. Springer International Publishing, 2018.

[2] Martin Bader and Enno Ohlebusch. Sorting by Weighted Reversals, Transpositions, and Inverted Transpositions. *Journal of Computational Biology*, 14(5):615–636, 2007.

[3] Vineet Bafna and Pavel A. Pevzner. Sorting by Transpositions. *SIAM Journal on Discrete Mathematics*, 11(2):224–240, 1998.

[4] Christian Baudet, Ulisses Dias, and Zanoni Dias. Length and Symmetry on the Sorting by Weighted Inversions Problem. In S. Campos, editor, *Advances in Bioinformatics and Computational Biology*, volume 8826 of *Lecture Notes in Computer Science*, pages 99–106. Springer International Publishing, Switzerland, 2014.

Referências

[5] Piotr Berman, Sridhar Hannenhalli, and Marek Karpinski. 1.375-Approximation Algorithm for Sorting by Reversals. In *Proceedings of the 10th Annual European Symposium on Algorithms (ESA'2002)*, volume 2461 of *Lecture Notes in Computer Science*, pages 200–210. Springer-Verlag Berlin Heidelberg New York, Berlin/Heidelberg, Germany, 2002.

[6] Mathieu Blanchette, Takashi Kunisawa, and David Sankoff. Parametric Genome Rearrangement. *Gene*, 172(1):GC11–GC17, 1996.

[7] Laurent Bulteau, Guillaume Fertin, and Irena Rusu. Sorting by Transpositions is Difficult. *SIAM Journal on Computing*, 26(3):1148–1180, 2012.

[8] Alberto Caprara. Sorting Permutations by Reversals and Eulerian Cycle Decompositions. *SIAM Journal on Discrete Mathematics*, 12(1):91–110, 1999.

[9] Xin Chen. On Sorting Unsigned Permutations by Double-Cut-and-Joins. *Journal of Combinatorial Optimization*, 25(3):339–351, 2013.

Referências

[10] Isaac Elias and Tzvika Hartman. A 1.375-Approximation Algorithm for Sorting by Transpositions. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(4):369–379, 2006.

[11] Niklas Eriksen. Combinatorics of Genome Rearrangements and Phylogeny. Teknologie Licentiat Thesis, Kungliga Tekniska Högskolan, Stockholm, 2001.

[12] Gustavo R. Galvão, Orlando Lee, and Zanoni Dias. Sorting Signed Permutations by Short Operations. *Algorithms for Molecular Biology*, 10(1):1–17, 2015.

[13] Sridhar Hannenhalli and Pavel A. Pevzner. Transforming Cabbage into Turnip: Polynomial Algorithm for Sorting Signed Permutations by Reversals. *Journal of the ACM*, 46(1):1–27, 1999.

[14] Lenwood S. Heath and John Paul C. Vergara. Sorting by Short Swaps. *Journal of Computational Biology*, 10(5):775–789, 2003.

Referências

[15] Mark R. Jerrum. The Complexity of Finding Minimum-length Generator Sequences. *Theoretical Computer Science*, 36(2-3):265–289, 1985.

[16] Haitao Jiang, Haodi Feng, and Daming Zhu. An 5/4-Approximation Algorithm for Sorting Permutations by Short Block Moves. In *Proceedings of the 25th International Symposium on Algorithms and Computation (ISAAC'2014)*, volume 8889 of *Lecture Notes in Computer Science*, pages 491–503. Springer International Publishing, 2014.

[17] Jean-François Lefebvre, Nadia El-Mabrouk, Elisabeth R. M. Tillier, and David Sankoff. Detection and validation of single gene inversions. *Bioinformatics*, 19(1):i190–i196, 2003.

[18] Guohui Lin and Tao Jiang. A Further Improved Approximation Algorithm for Breakpoint Graph Decomposition. *Journal of Combinatorial Optimization*, 8(2):183–194, 2004.

[19] Carla N. Lintzmayer, Guillaume Fertin, and Zanoni Dias. Approximation Algorithms for Sorting by Length-Weighted Prefix and Suffix Operations. *Theoretical Computer Science*, 593:26–41, 2015.

[20] Carla Negri Lintzmayer, Guillaume Fertin, and Zanoni Dias. Sorting permutations and binary strings by length-weighted rearrangements. *Theoretical Computer Science*, 715:35–59, 2018.

[21] Guilherme Henrique Santos Miranda, Alexsandro Oliveira Alexandrino, Carla Negri Lintzmayer, and Zanoni Dias. Sorting λ-permutations by λ-operations. In *Advances in Bioinformatics and Computational Biology*, pages 1–13. Springer International Publishing, 2018.

[22] Thach C. Nguyen, Hieu T. Ngo, and Nguyen B. Nguyen. Sorting by Restricted-Length-Weighted Reversals. *Genomics Proteomics & Bioinformatics*, 3(2):120–127, 2005.

[23] Ron Y. Pinter and Steven Skiena. Genomic Sorting with Length-Weighted Reversals. *Genome Informatics*, 13:2002, 2002.

Referências

[24] Atif Rahman, Swakkhar Shatabda, and Masud Hasan. An Approximation Algorithm for Sorting by Reversals and Transpositions. *Journal of Discrete Algorithms*, 6(3):449–457, 2008.

[25] John P. C. Vergara. *Sorting by Bounded Permutations*. PhD thesis, Virginia Polytechnic Institute and State University, 1998.

[26] Maria E. M. T. Walter, Zanoni Dias, and João Meidanis. Reversal and Transposition Distance of Linear Chromosomes. In *Proceedings of the 5th International Symposium on String Processing and Information Retrieval (SPIRE'1998)*, pages 96–102, Los Alamitos, CA, USA, 1998. IEEE Computer Society.

Resultados Experimentais: Operações Ponderadas pelo Número de Fragmentações

Resultados Experimentais

- Instâncias:
  - UB_n: 1000 permutações sem sinais aleatórias de tamanho n com número máximo de *breakpoints*;
  - SB_n: 1000 permutações com sinais aleatórias de tamanho n com número máximo de *breakpoints*;
  - R_n^M: 1000 permutações de tamanho n geradas a partir da aplicação de $n/5$ operações aleatórias na permutação identidade ι.

Resultados Experimentais

- Algoritmos Adaptados:
  - A-R: $(1.4193 + \epsilon)$-aproximação para a Ordenação de Permutações por Reversões sem Sinais [13, 18];
  - A-$\bar{\text{R}}$: algoritmo exato para a Ordenação de Permutações por Reversões com Sinais [13];
  - A-T: 1.5-aproximação para a Ordenação de Permutações por Transposições [3];
  - A-RT: $2k$-aproximação, com $k = 1.4193 + \epsilon$ [18], para a Ordenação de Permutações por Reversões sem Sinais e Transposições [24];
  - A-$\bar{\text{R}}$T: 2-aproximação para a Ordenação de Permutações por Reversões com Sinais e Transposições [24].

Resultados Experimentais

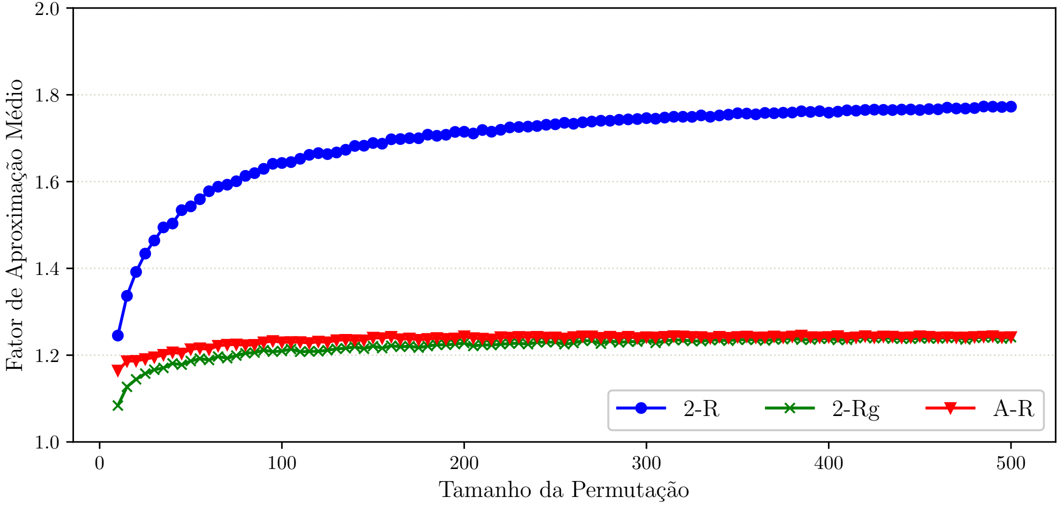

Figura 4: Fator de aproximação médio para os algoritmos 2-R, 2-Rg e A-R e instâncias R_n^r.

Resultados Experimentais

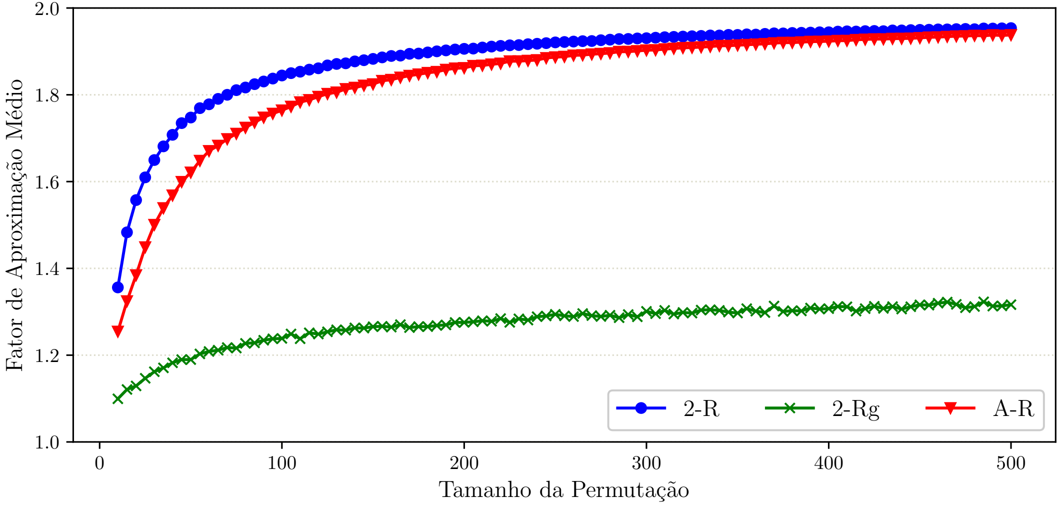

Figura 5: Fator de aproximação médio para os algoritmos 2-R, 2-Rg e A-R e instâncias UB_n.

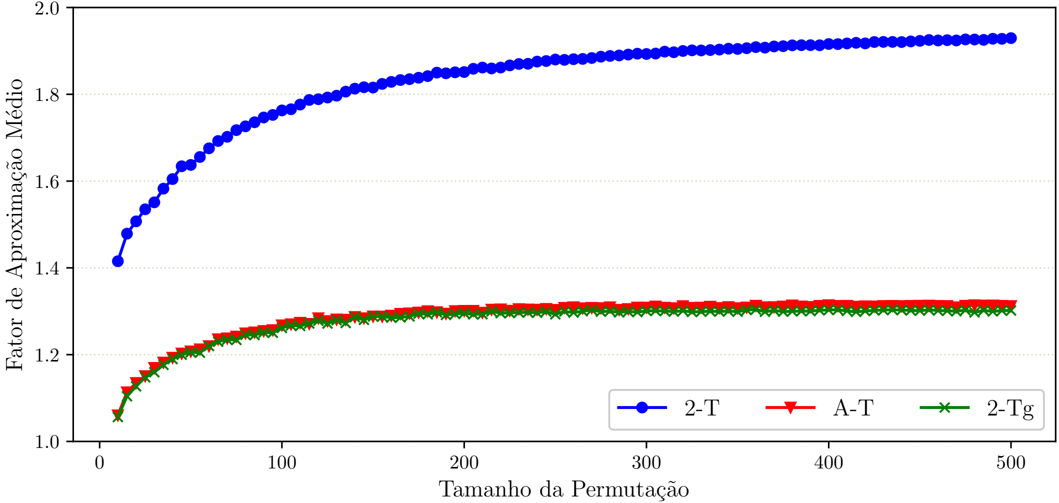

Figura 6: Fator de aproximação médio para os algoritmos 2-T, 2-Tg e A-T e instâncias R_n^t.

Resultados Experimentais

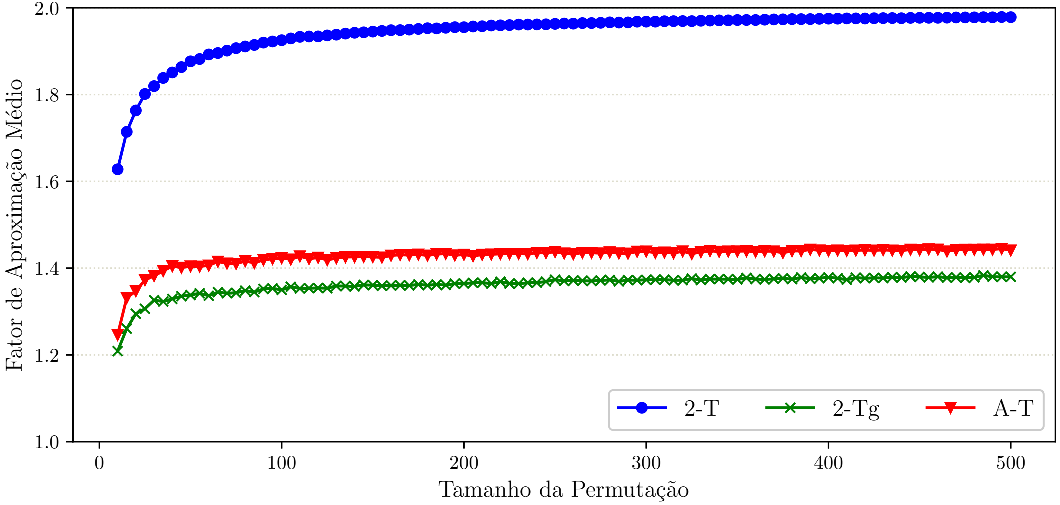

Figura 7: Fator de aproximação médio para os algoritmos 2-T, 2-Tg e A-T e instâncias UB_n.

Resultados Experimentais

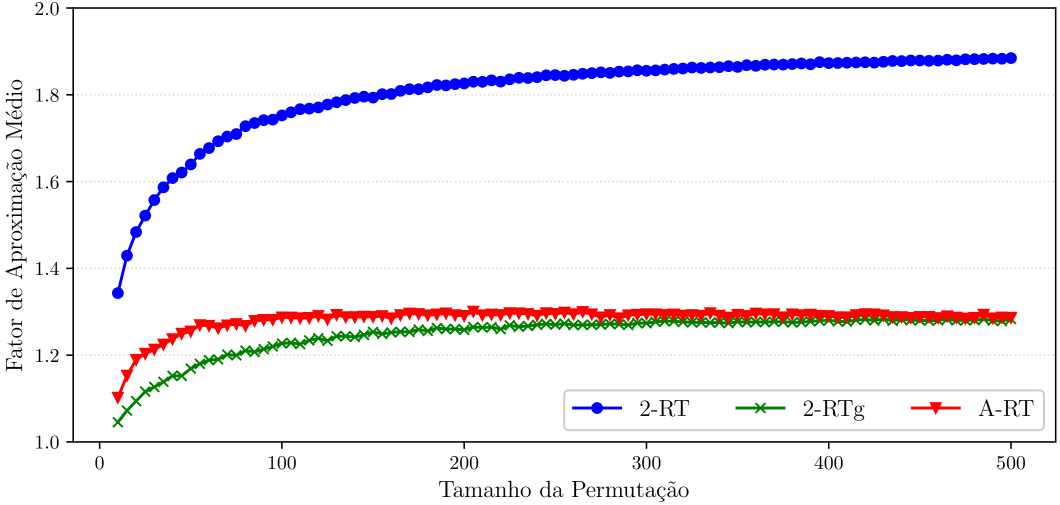

Figura 8: Fator de aproximação médio para os algoritmos 2-RT, 2-RTg e A-RT e instâncias R_n^{rt}.

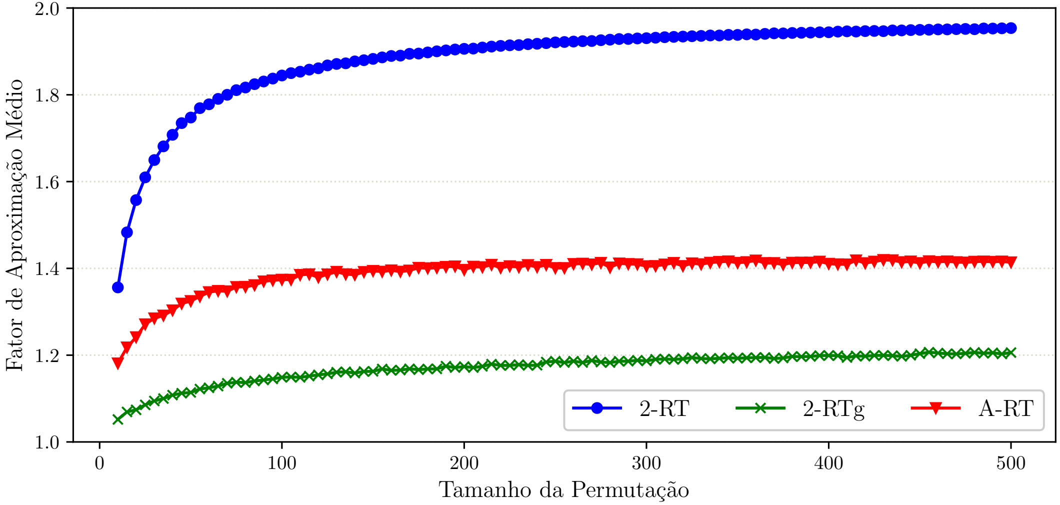

Figura 9: Fator de aproximação médio para os algoritmos 2-RT, 2-RTg e A-RT e instâncias UB_n.

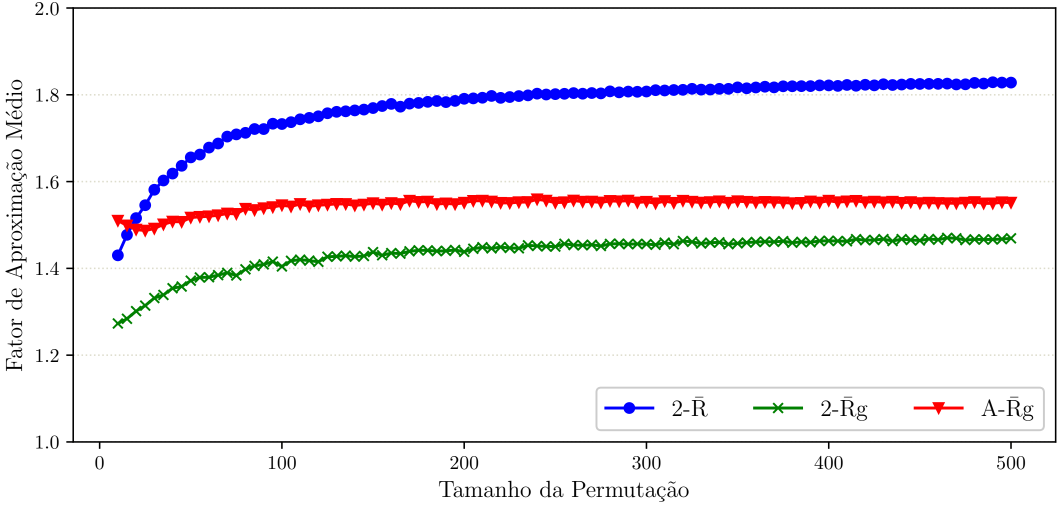

Figura 10: Fator de aproximação médio para os algoritmos 2-R̄, 2-R̄g e A-R̄ e instâncias $\mathrm{R}_n^{\bar{r}}$.

Resultados Experimentais

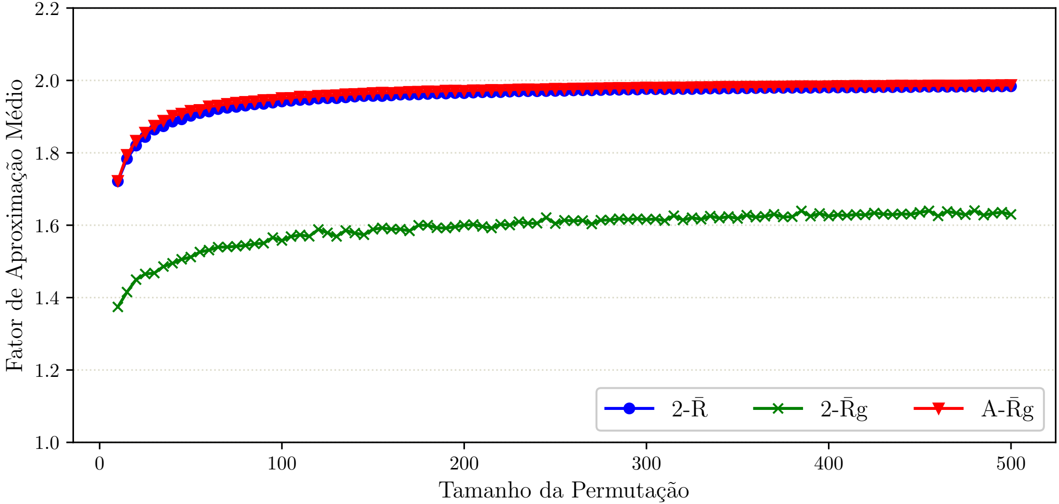

Figura 11: Fator de aproximação médio para os algoritmos 2-R̄, 2-R̄g e A-R̄ e instâncias SB_n.

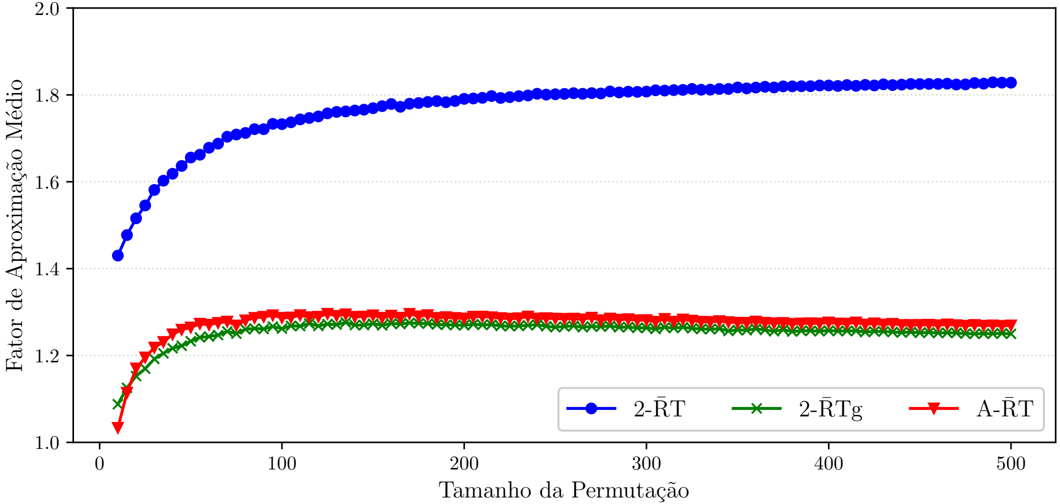

Figura 12: Fator de aproximação médio para os algoritmos 2-R̄T, 2-R̄Tg e A-R̄T e instâncias $\mathrm{R}_n^{\bar{r}t}$.

Resultados Experimentais

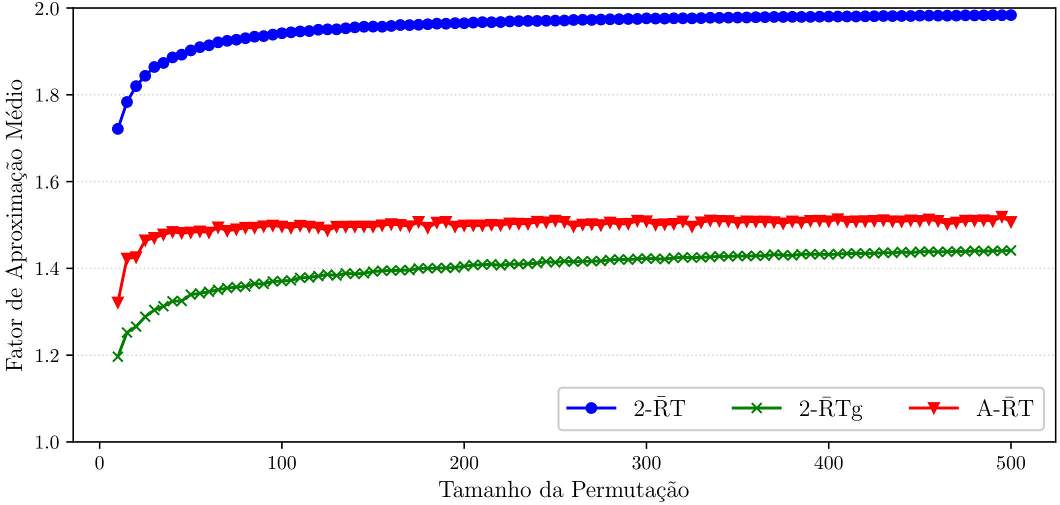

Figura 13: Fator de aproximação médio para os algoritmos 2-R̄T, 2-R̄Tg e A-R̄T e instâncias SB_n.

Resultados Experimentais: Operações Curtas Ponderadas pelo Tamanho

Resultados Experimentais

- Instâncias:
  - $\mathrm{SR01}_n^M$: 1000 permutações de tamanho n geradas a partir da aplicação de $\lceil \sqrt{n} \rceil$ operações curtas aleatórias na permutação identidade ι;
  - $\mathrm{SR02}_n^M$: 1000 permutações de tamanho n geradas a partir da aplicação de $\lceil \frac{n}{2} \rceil$ operações curtas aleatórias na permutação identidade ι;
  - $\mathrm{SR03}_n^M$: 1000 permutações de tamanho n geradas a partir da aplicação de n operações curtas aleatórias na permutação identidade ι;
  - $\mathrm{SR04}_n^M$: 1000 permutações de tamanho n geradas a partir da aplicação de n^2 operações curtas aleatórias na permutação identidade ι.

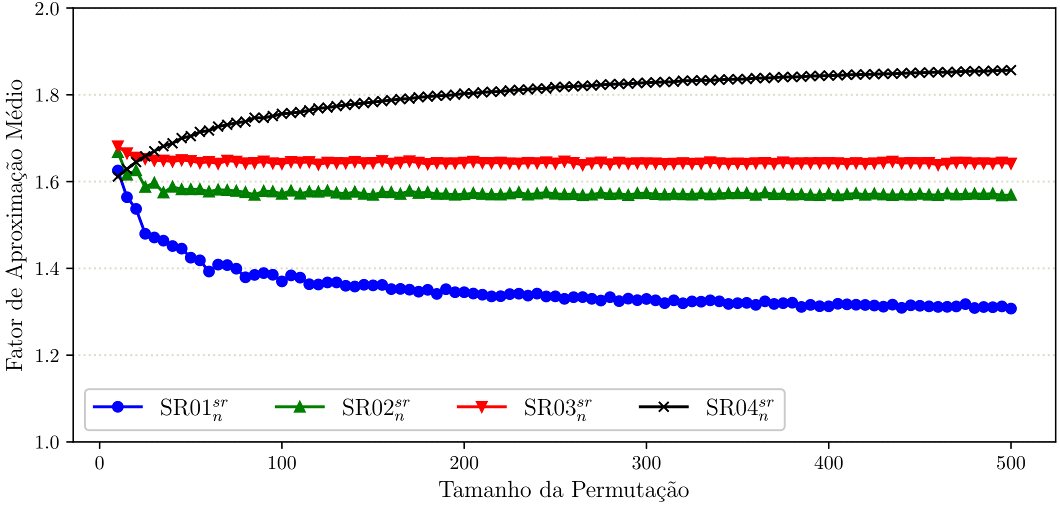

Figura 14: Fator de aproximação médio para o algoritmo 2-sR.

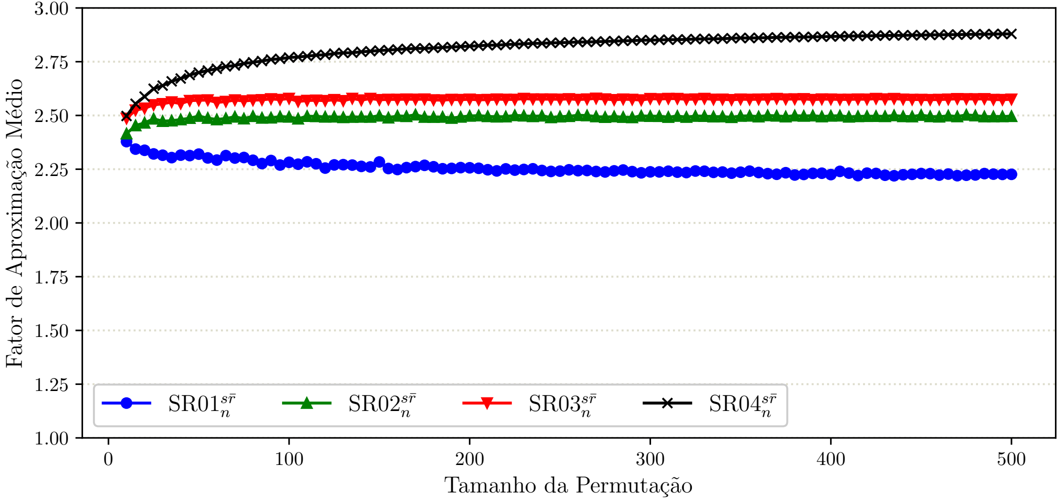

Figura 15: Fator de aproximação médio para o algoritmo 3-s$\bar{\text{R}}$.

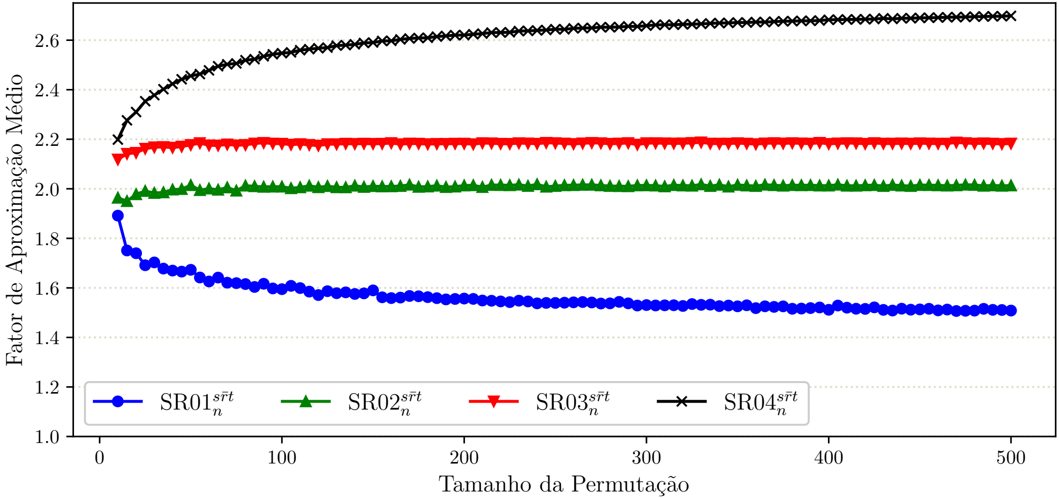

Figura 16: Fator de aproximação médio para o algoritmo 3-s$\bar{\text{R}}$sT.

Resultados Experimentais

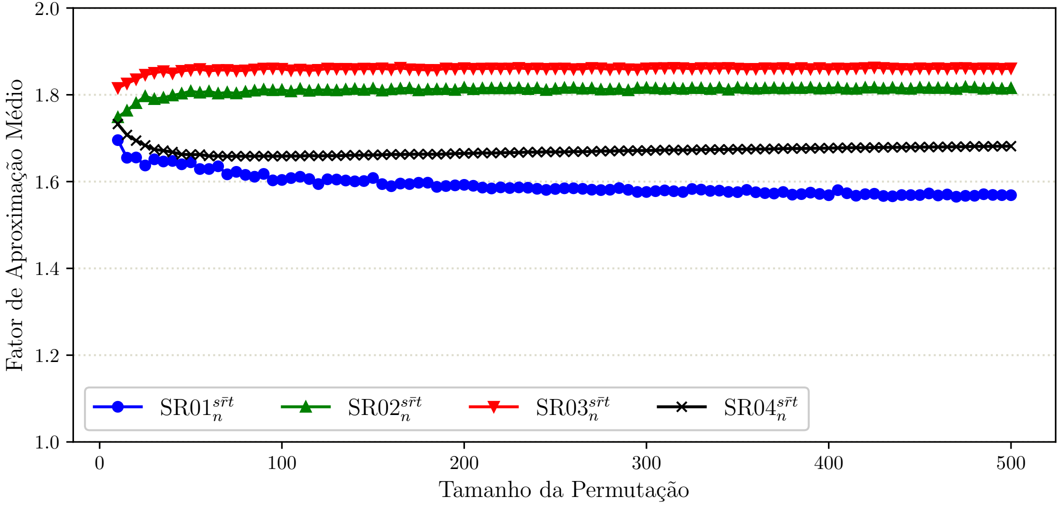

Figura 17: Fator de aproximação médio para o algoritmo 7/3-s$\bar{\text{R}}$sT.

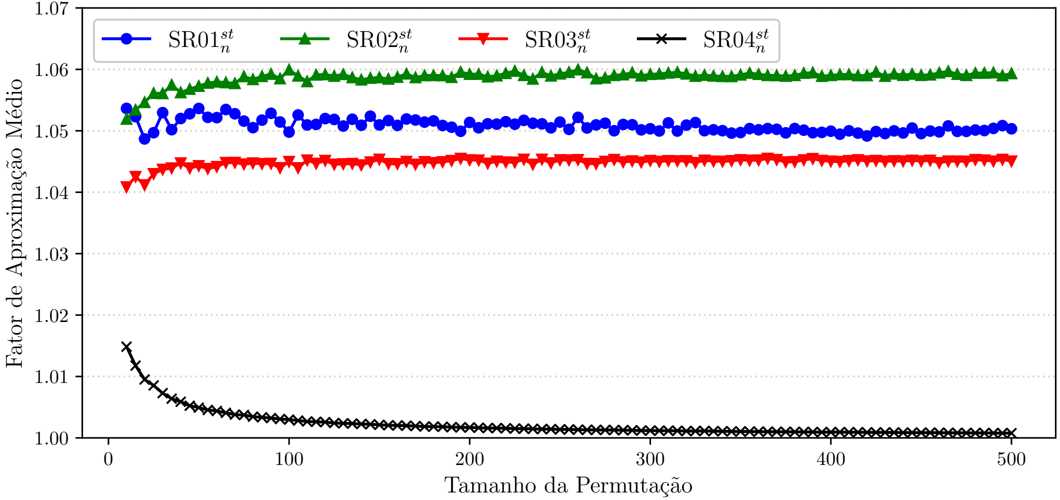

Figura 18: Fator de aproximação médio para o algoritmo 4/3-sT.

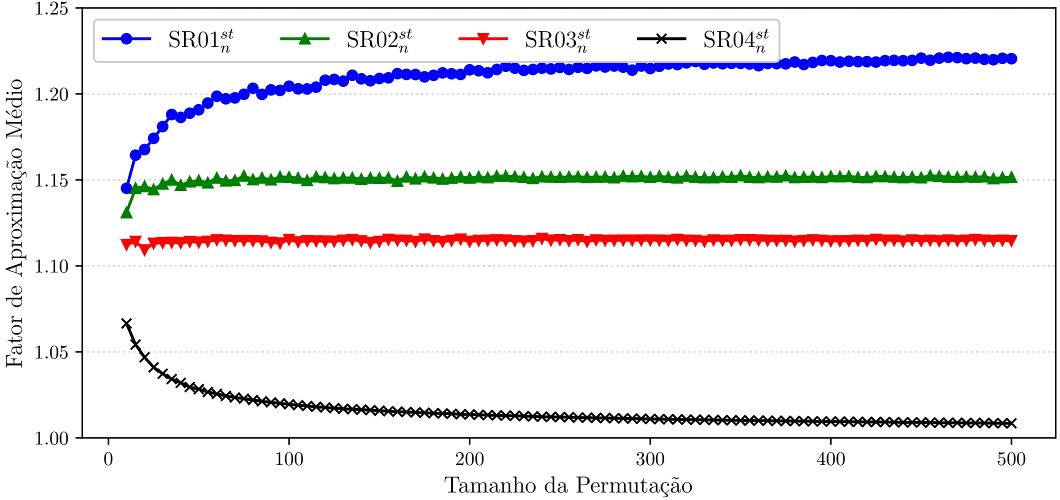

Figura 19: Fator de aproximação médio para o algoritmo ManyInversions-sT.

Resultados Experimentais

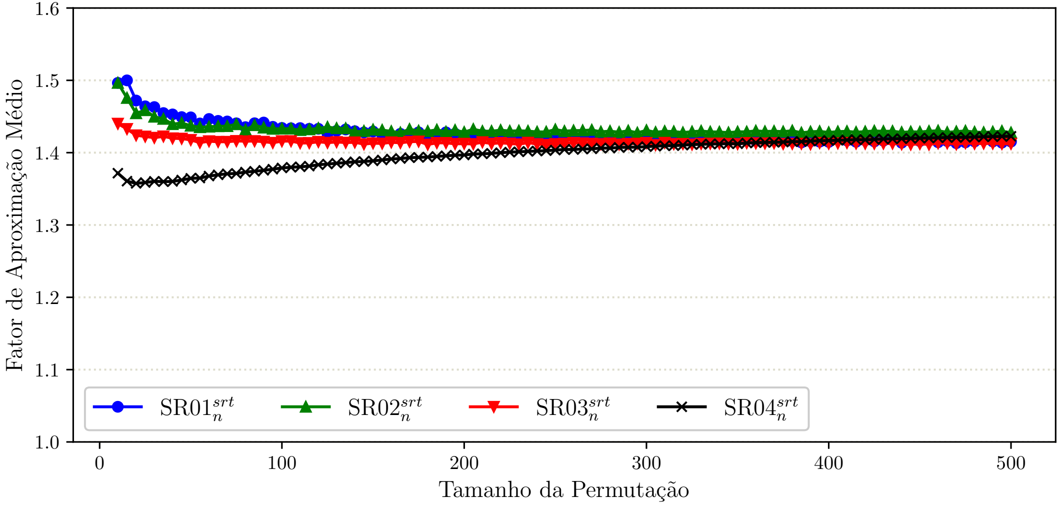

Figura 20: Fator de aproximação médio para o algoritmo 2-sRsT.