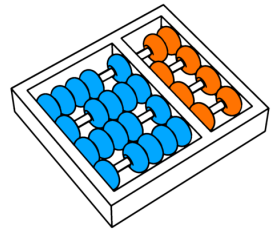


Ordenação de Permutações com Sinais por Reversões e Transposições

Klairton de Lima Brito

Orientador: Zanoni Dias

Instituto de Computação
Universidade Estadual de Campinas
Dissertação de Mestrado

Roteiro

1. Motivação
2. Conceitos
3. Heurísticas
4. Algoritmos de Aproximação
5. Conclusões

Motivação

Motivação

- Biologia Computacional.
- Comparação de Genomas.
- Rearranjo de Genomas.

Conceitos

Conceitos

Representação de um Genoma

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

Permutações com sinais

- $\pi = (+5\ +2\ +4\ -1\ -3)$
- $\iota_n = (+1\ \ldots\ +n)$
- $\eta_n = (-n\ \ldots\ -1)$

Conceitos

Composição

- A composição entre as permutações π e σ, definida como $\pi \circ \sigma$, é uma operação que resulta em uma nova permutação α.

$$\begin{aligned} \pi &= (\pi_1\ \pi_2\ \ldots\ \pi_n) \\ \sigma &= (\sigma_1\ \sigma_2\ \ldots\ \sigma_n) \\ \pi \circ \sigma &= (\pi_{\sigma_1}\ \pi_{\sigma_2}\ \ldots\ \pi_{\sigma_n}) = \alpha \end{aligned}$$

$$\alpha_i = \begin{cases} -\pi_{|\sigma_i|}, & \text{se } \sigma_i < 0 \\ \pi_{\sigma_i}, & \text{se } \sigma_i > 0 \end{cases}$$

Exemplo

$$\begin{aligned} \pi &= (-3\ +2\ -5\ +1\ +4\ +6) \\ \sigma &= (+1\ -4\ -3\ -2\ +5\ +6) \\ \pi \circ \sigma &= (-3\ -1\ +5\ -2\ +4\ +6) = \alpha \end{aligned}$$

Conceitos

Modelo de Rearranjo

- Define quais eventos podem ser aplicados para transformar uma permutação em outra.
- A *distância* entre as permutações π e σ no modelo M é o tamanho da menor sequência de eventos de rearranjo que transforma π em σ, e é denotada por $d_M(\pi, \sigma)$.

Eventos Abordados

- Reversão.
- Transposição.
- Variações com restrições adicionais.

Distância $\times$ Ordenação

Permutação Inversa

- A permutação inversa de uma permutação π é representada como π^{-1}. Essa permutação indica a posição e a orientação de cada elemento i em π.

$$\begin{aligned}
\pi &= (+2\ -1\ +4\ -5\ +3\ +6) \\
\pi^{-1} &= (-2\ +1\ +5\ +3\ -4\ +6) \\
\pi \circ \pi^{-1} &= \pi^{-1} \circ \pi = \iota_6
\end{aligned}$$

Equivalência

$$\begin{aligned}
& d_M(\pi, \sigma) \\
&= d_M(\pi \circ \sigma^{-1}, \sigma \circ \sigma^{-1}) \\
&= d_M(\alpha, \iota_n)
\end{aligned}$$

Reversão

Definições

- Uma reversão $\rho(i, j)$, com $1 \leq i \leq j \leq n$, é um evento que inverte a posição e a orientação dos genes em um bloco do genoma.

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

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

- Uma reversão de prefixo $\rho(1, j)$, com $1 \leq j \leq n$, é uma reversão que afeta um bloco no início do genoma.
- Uma reversão de sufixo $\rho(i, n)$, com $1 \leq i \leq n$, é uma reversão que afeta um bloco no fim do genoma.

Transposição

Definições

- Uma transposição $\tau(i, j, k)$, com $1 \leq i < j < k \leq n + 1$, é um evento que troca de posição dois blocos adjacentes de um mesmo genoma $\pi[i \ldots j-1]$ e $\pi[j \ldots k-1]$, mas sem afetar a orientação e a posição dos genes nos blocos.

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

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

- Uma transposição de prefixo $\tau(1, j, k)$, com $1 < j < k \leq n + 1$, é uma transposição que afeta um bloco no início do genoma.
- Uma transposição de sufixo $\tau(i, j, n + 1)$, com $1 \leq i < j < n + 1$, é uma transposição que afeta um bloco no fim do genoma.

Breakpoints

A permutação estendida é obtida a partir de π inserindo dois novos elementos: $\pi_0 = +0$ e $\pi_{n+1} = +(n+1)$.

Definições

- Um breakpoint ocorre entre um par de elementos π_i e π_{i+1} de π, com $0 \leq i \leq n$, se $\pi_{i+1} - \pi_i \neq 1$.
- Um breakpoint de prefixo ocorre em um par de elementos π_i e π_{i+1} de π, com $0 < i \leq n$, tal que $\pi_{i+1} - \pi_i \neq 1$.
- Um breakpoint de prefixo ou sufixo ocorre em um par de elementos π_i e π_{i+1} de π, com $0 < i < n$, tal que $\pi_{i+1} - \pi_i \neq 1$.

Exemplo de Breakpoint

$$\pi = (+0 \cdot -2 \ -1 \cdot +4 \ +5 \cdot -3 \cdot +6)$$

Breakpoints

A permutação estendida é obtida a partir de π inserindo dois novos elementos: $\pi_0 = +0$ e $\pi_{n+1} = +(n+1)$.

Definições

- Um breakpoint ocorre entre um par de elementos π_i e π_{i+1} de π, com $0 \leq i \leq n$, se $\pi_{i+1} - \pi_i \neq 1$.
- Um breakpoint de prefixo ocorre em um par de elementos π_i e π_{i+1} de π, com $0 < i \leq n$, tal que $\pi_{i+1} - \pi_i \neq 1$.
- Um breakpoint de prefixo ou sufixo ocorre em um par de elementos π_i e π_{i+1} de π, com $0 < i < n$, tal que $\pi_{i+1} - \pi_i \neq 1$.

Exemplo de Breakpoint de Prefixo

$$\pi \ = \ (+0 \ -2 \ -1 \cdot +4 \ +5 \cdot -3 \cdot +6)$$

Breakpoints

A permutação estendida é obtida a partir de π inserindo dois novos elementos: $\pi_0 = +0$ e $\pi_{n+1} = +(n+1)$.

Definições

- Um breakpoint ocorre entre um par de elementos π_i e π_{i+1} de π, com $0 \leq i \leq n$, se $\pi_{i+1} - \pi_i \neq 1$.
- Um breakpoint de prefixo ocorre em um par de elementos π_i e π_{i+1} de π, com $0 < i \leq n$, tal que $\pi_{i+1} - \pi_i \neq 1$.
- Um breakpoint de prefixo ou sufixo ocorre em um par de elementos π_i e π_{i+1} de π, com $0 < i < n$, tal que $\pi_{i+1} - \pi_i \neq 1$.

Exemplo de Breakpoint de Prefixo ou Sufixo

$$\pi = (+0\ -2\ -1 \cdot +4\ +5 \cdot -3\ +6)$$

Grafo de Ciclos

O grafo de ciclos [3] $G(\pi)$ é construído a partir de uma permutação com sinais π. O grafo é formado por um conjunto de vértices $V(\pi)$, um conjunto de arestas pretas $E_p(\pi)$ e um conjunto de arestas cinzas $E_c(\pi)$ de forma que:

- $V(\pi) = \{+\pi_0, -\pi_1, +\pi_1, \ldots, -\pi_n, +\pi_n, -\pi_{n+1}\}$
- $E_p(\pi) = \bigcup_{i=1}^{n+1} \{(-\pi_i, +\pi_{i-1})\}$
- $E_c(\pi) = \bigcup_{i=1}^{n+1} \{(+(i-1), -i)\}$

Grafo de Ciclos

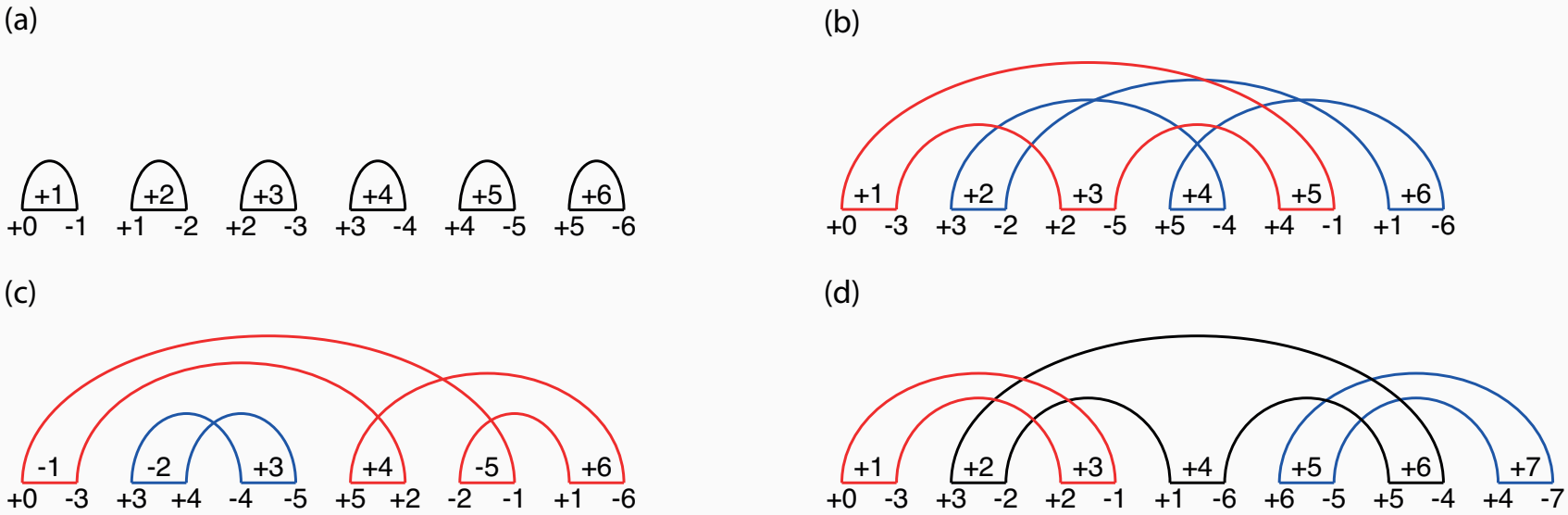


Figura 1: Exemplos de grafo de ciclos.

Permutação Simples

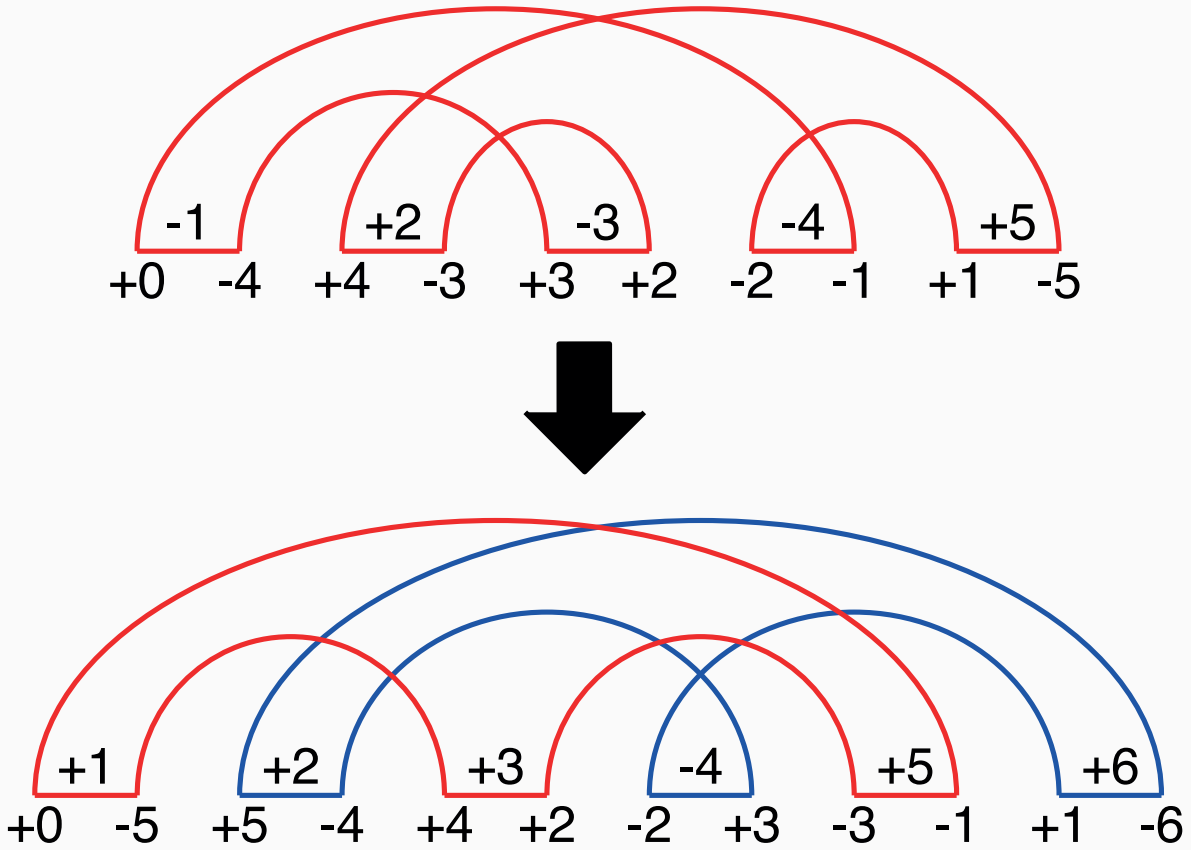


Figura 2: Obtendo a permutação simples $\pi' = (+5\ +4\ -2\ -3\ +1)$ a partir da permutação $\pi = (+4\ +3\ -2\ +1)$.

Heurísticas

Sliding Window

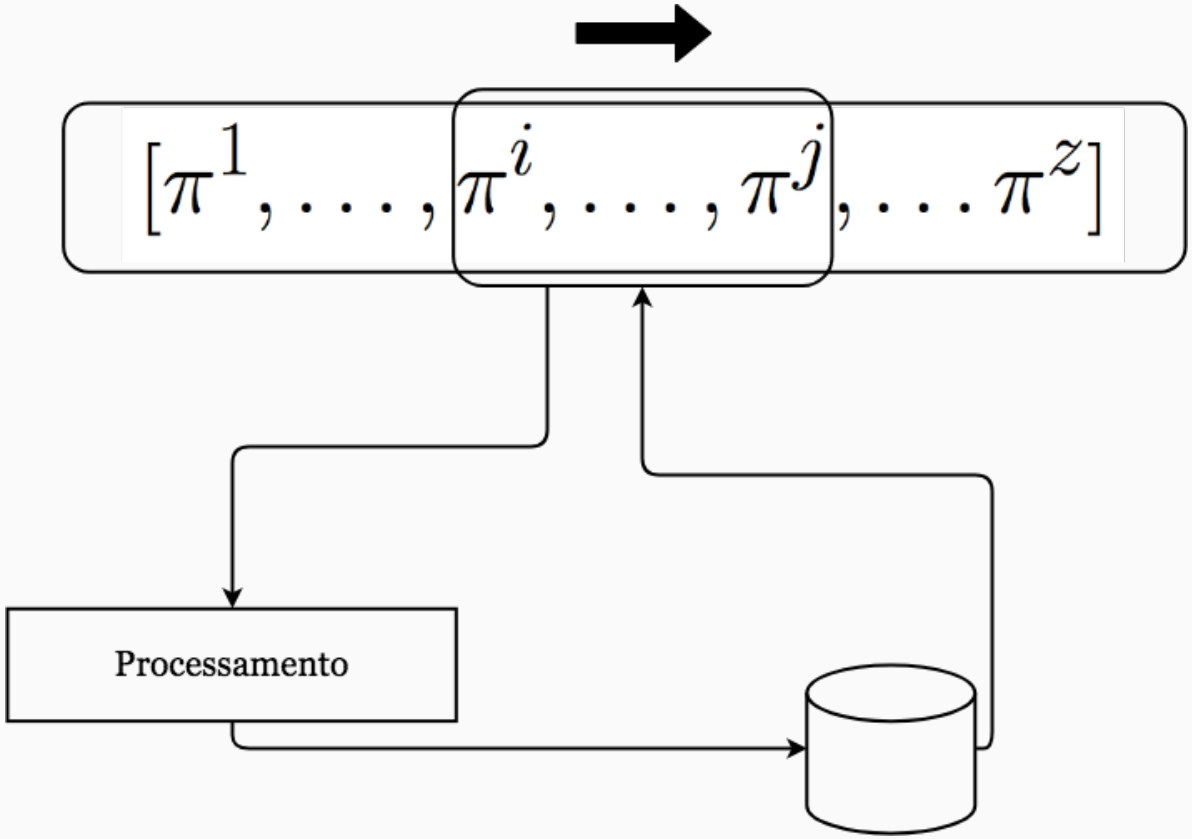


Figura 3: Exemplo de funcionamento da heurística Sliding Window.

Look Ahead

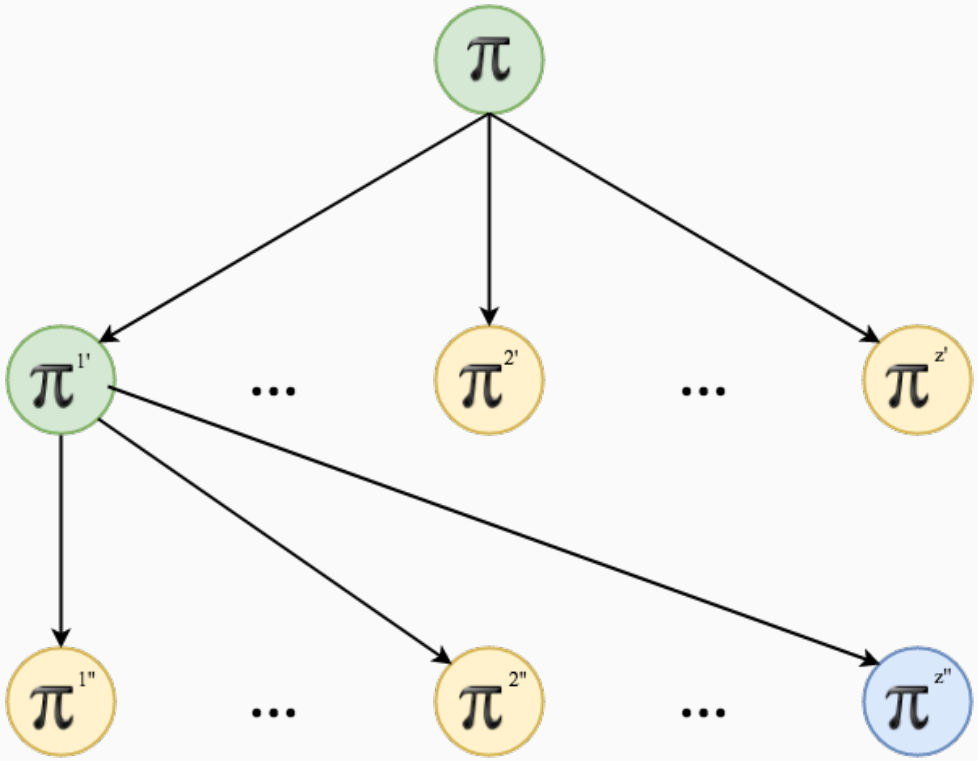


Figura 4: Exemplo de funcionamento da heurística Look Ahead.

Iterative Sliding Window

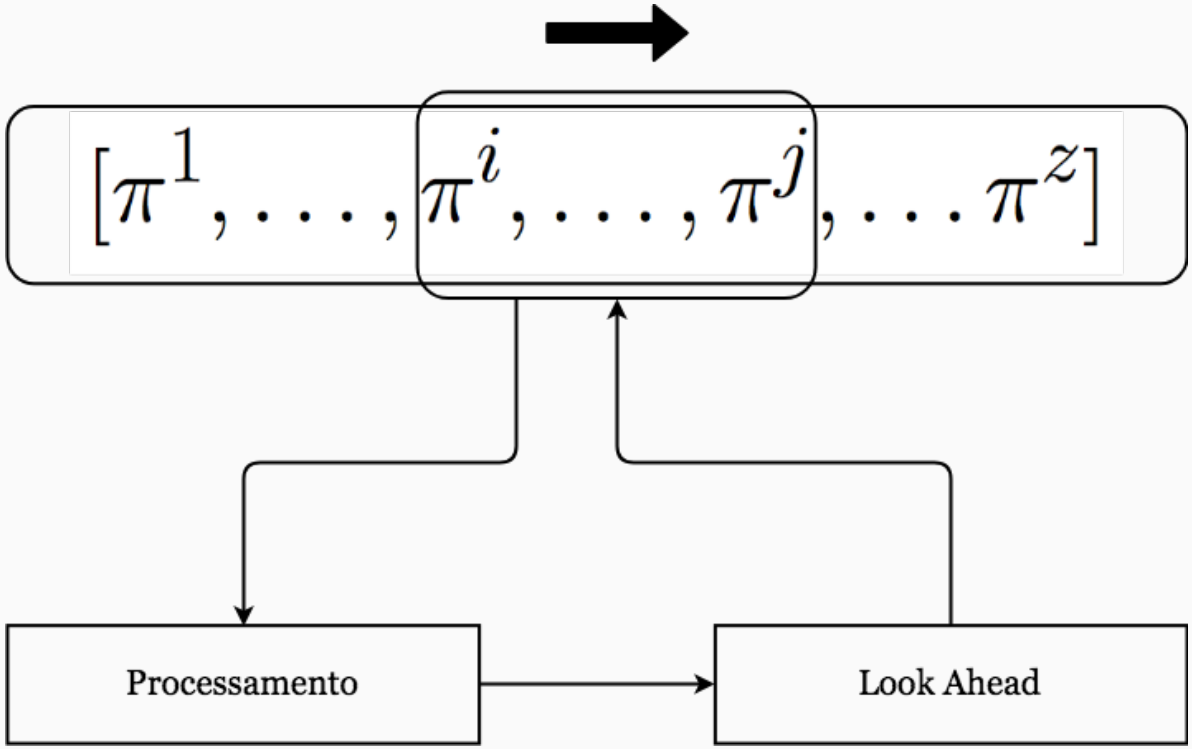


Figura 5: Exemplo de funcionamento da heurística Iterative Sliding Window.

Algoritmos Utilizados pelas Heurísticas

Tabela 1: Algoritmos clássicos utilizados.

Problemas Clássicos	**Código**	**Referência**	**Complexidade**	**Aproximação**
Reversão e Transposição	RSH	[7]	$O(n^3)$	$2k$ / 2
Reversão e Transposição com Sinal	WDM	[8]	$O(n^3)$	2 / 2
	BRPT	[8]	$O(n^2)$	3 / 3
	BRPR	[8]	$O(n^2)$	3 / 3
Reversão com Sinal	HPB	[5]	$O(n^2)$	1 / 3
		[1]	$O(n)$	1 / 3
Transposição	BP	[3]	$O(n^2)$	1.5 / $O(n)$

Algoritmos Utilizados pelas Heurísticas

Tabela 2: Algoritmos de prefixo utilizados.

Problemas de Prefixo	**Código**	**Referência**	**Complexidade**	**Aproximação**
Reversão com Sinal	2-SPR	[6]	$O(n^2)$	2 / 6
Reversão e Transposição com Sinal	2-SPRT 4-SPRT	[6]	$O(n^2)$ $O(n^2)$	$2+\lambda$ / $2+\lambda$ 4 / 4

Tabela 3: Algoritmos de prefixo ou sufixo utilizados.

Problemas de Prefixo ou Sufixo	**Código**	**Referência**	**Complexidade**	**Aproximação**
Reversão com Sinal	2-SPSR	[6]	$O(n^2)$	2 / 6
Reversão e Transposição com Sinal	2-SPSRT	[6]	$O(n^2)$	$2+\lambda$ / $2+\lambda$

Conjunto de Dados

- Um total de 18 conjuntos de permutações.
- Tamanhos $10, 20, \ldots, 100$ e $150, 200, \ldots, 500$.
- Cada conjunto com 1000 permutações.
- Maior número possível de breakpoints.

Sliding Window

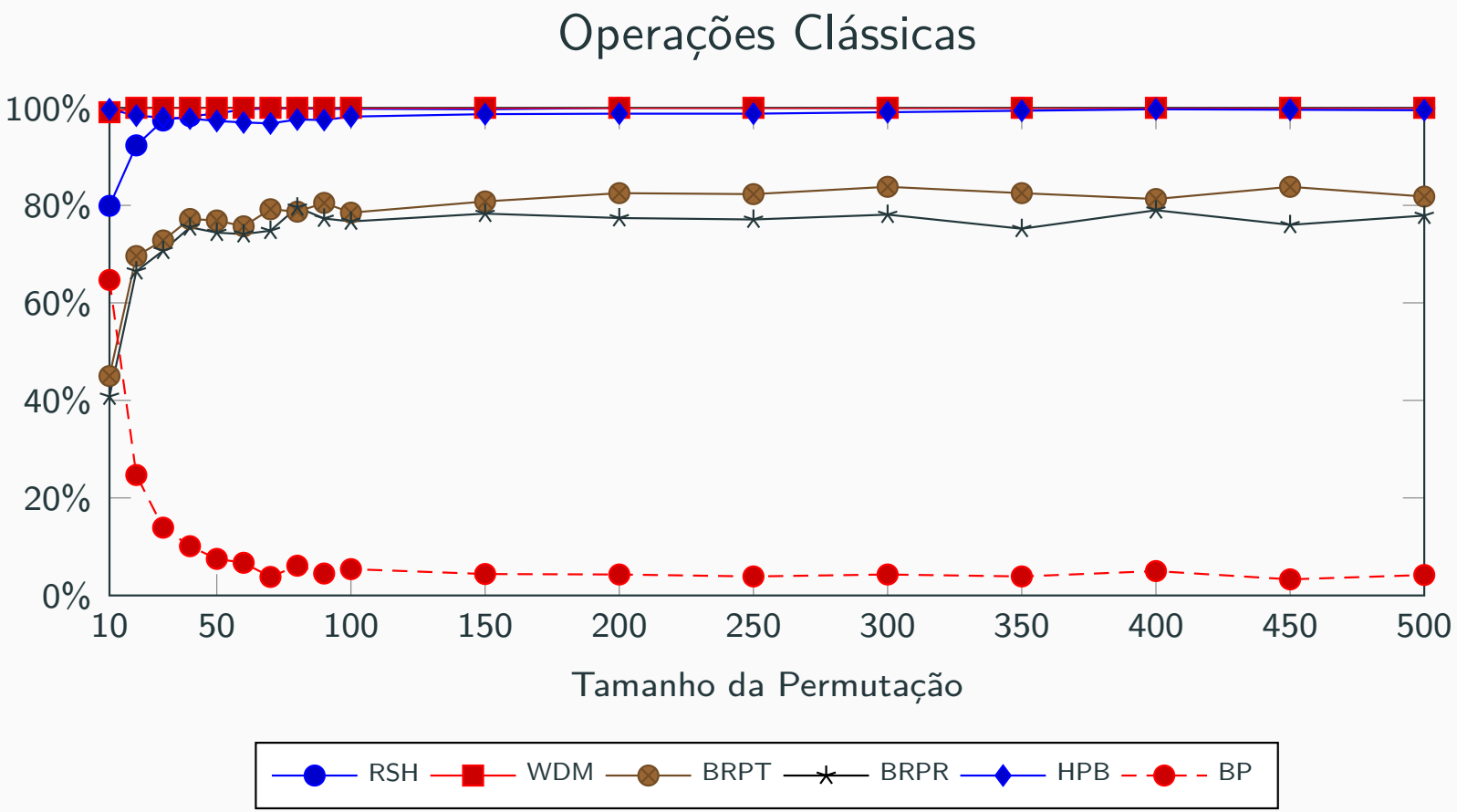


Figura 6: Porcentagem de soluções melhoradas utilizando Sliding Window.

Sliding Window

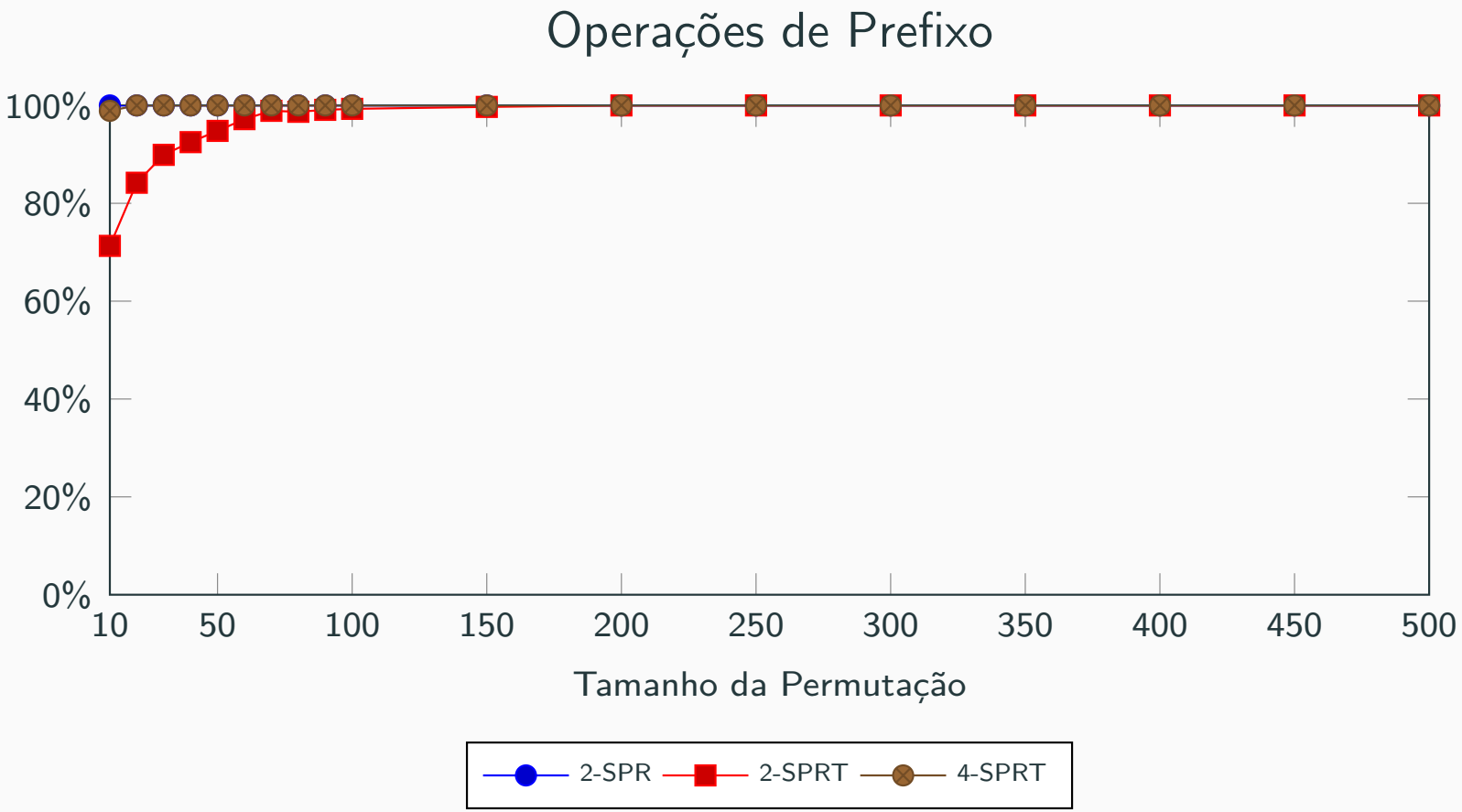


Figura 7: Porcentagem de soluções melhoradas utilizando Sliding Window.

Sliding Window

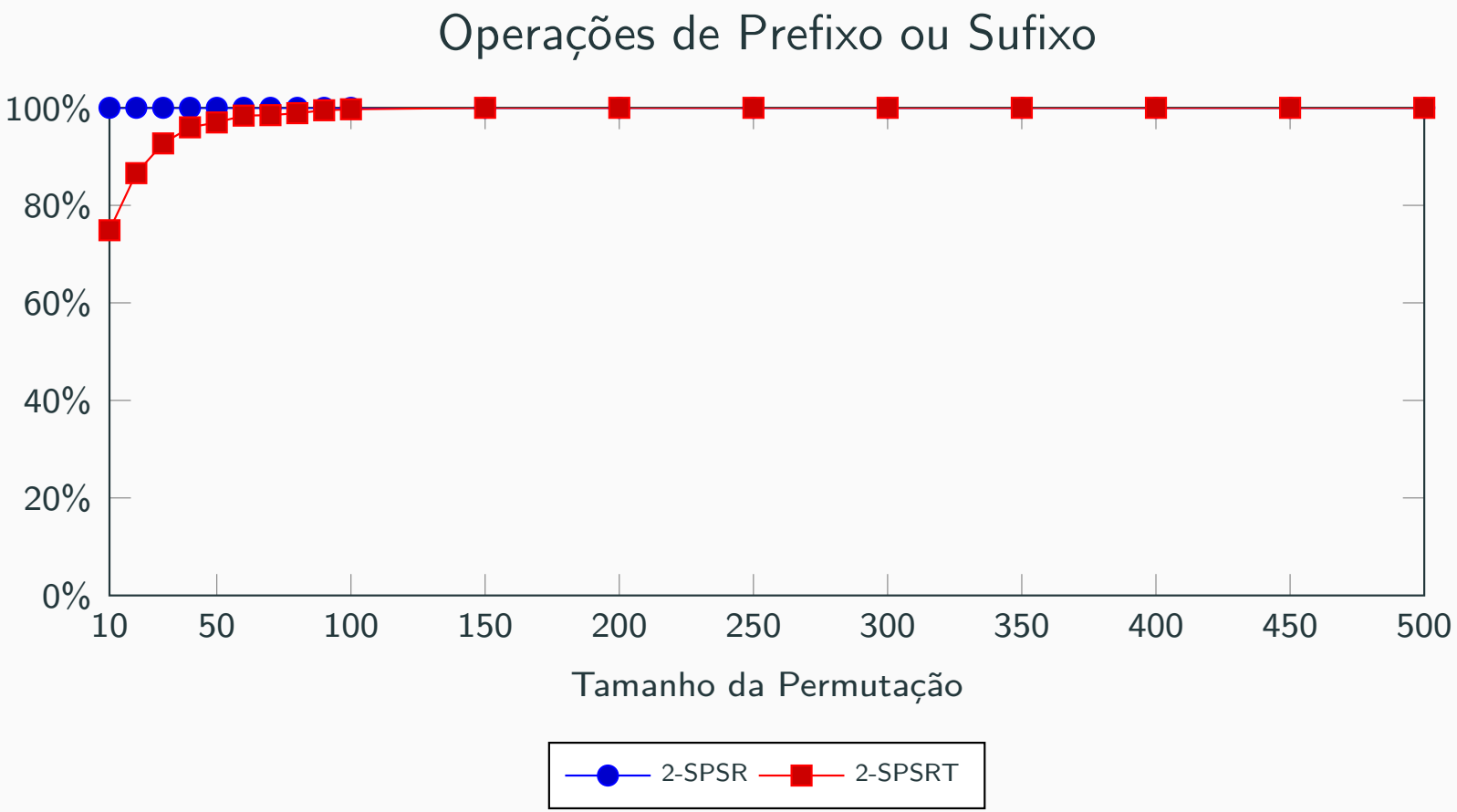


Figura 8: Porcentagem de soluções melhoradas utilizando Sliding Window.

Look Ahead

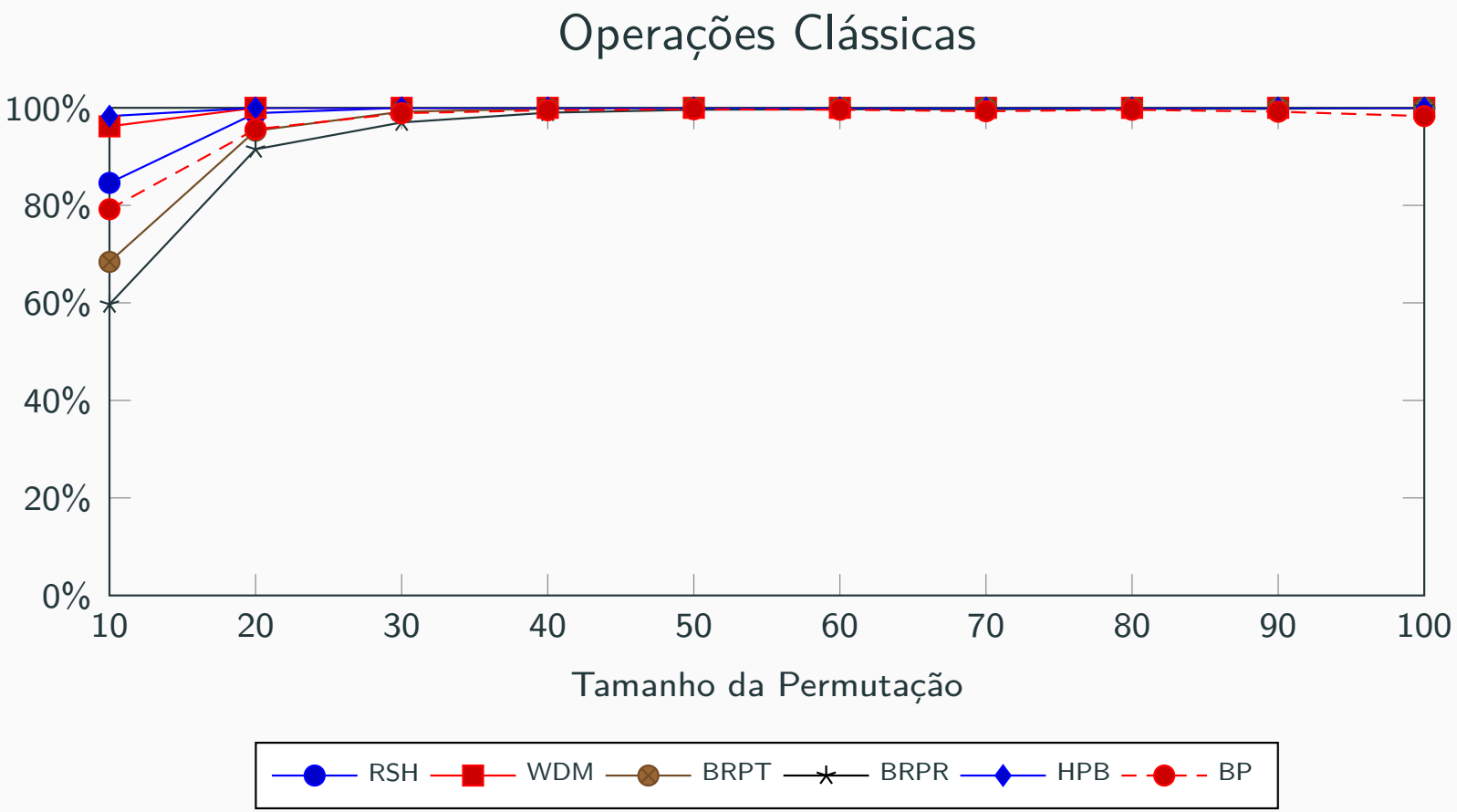


Figura 9: Porcentagem de soluções melhoradas utilizando Look Ahead.

Look Ahead

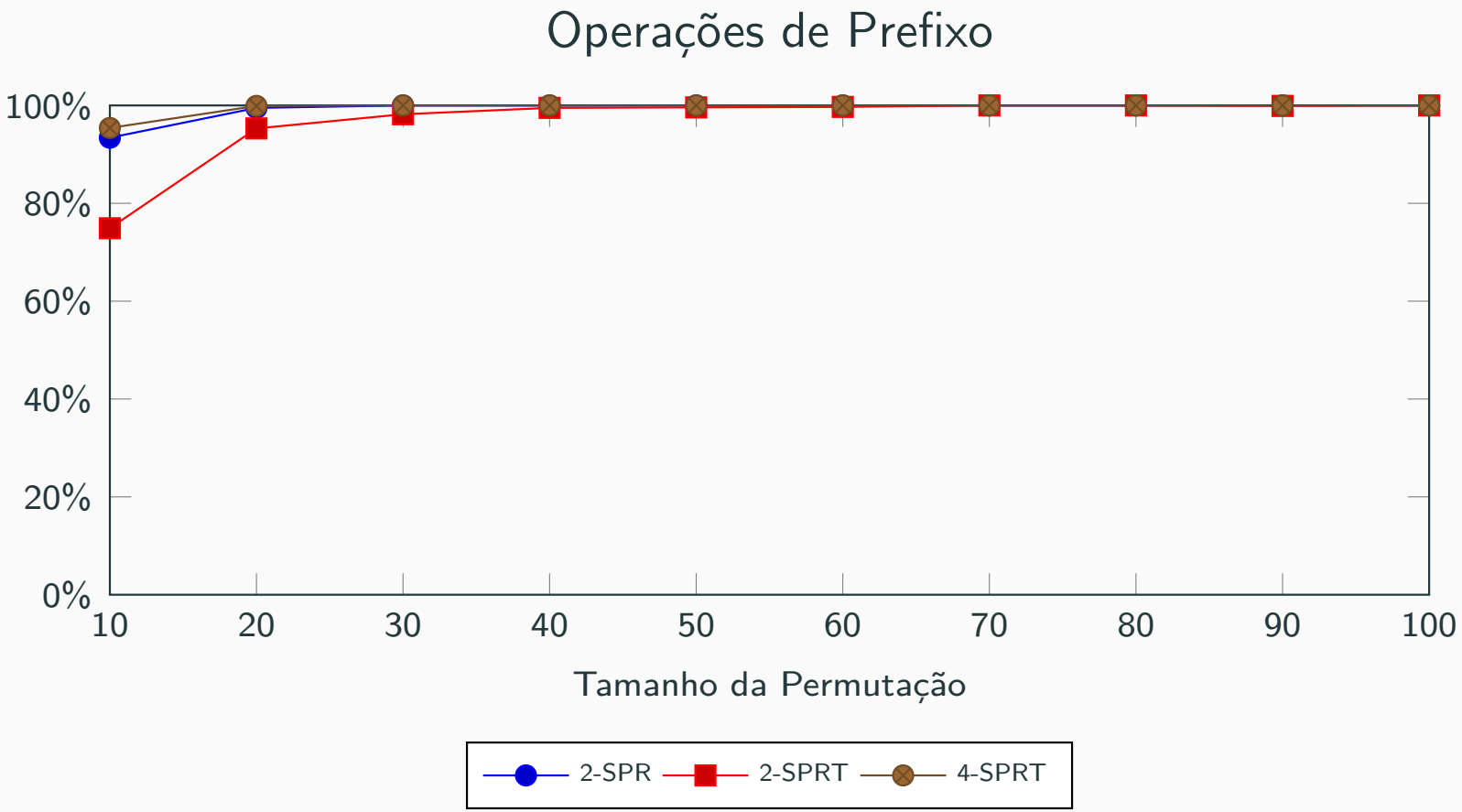


Figura 10: Porcentagem de soluções melhoradas utilizando Look Ahead.

Look Ahead

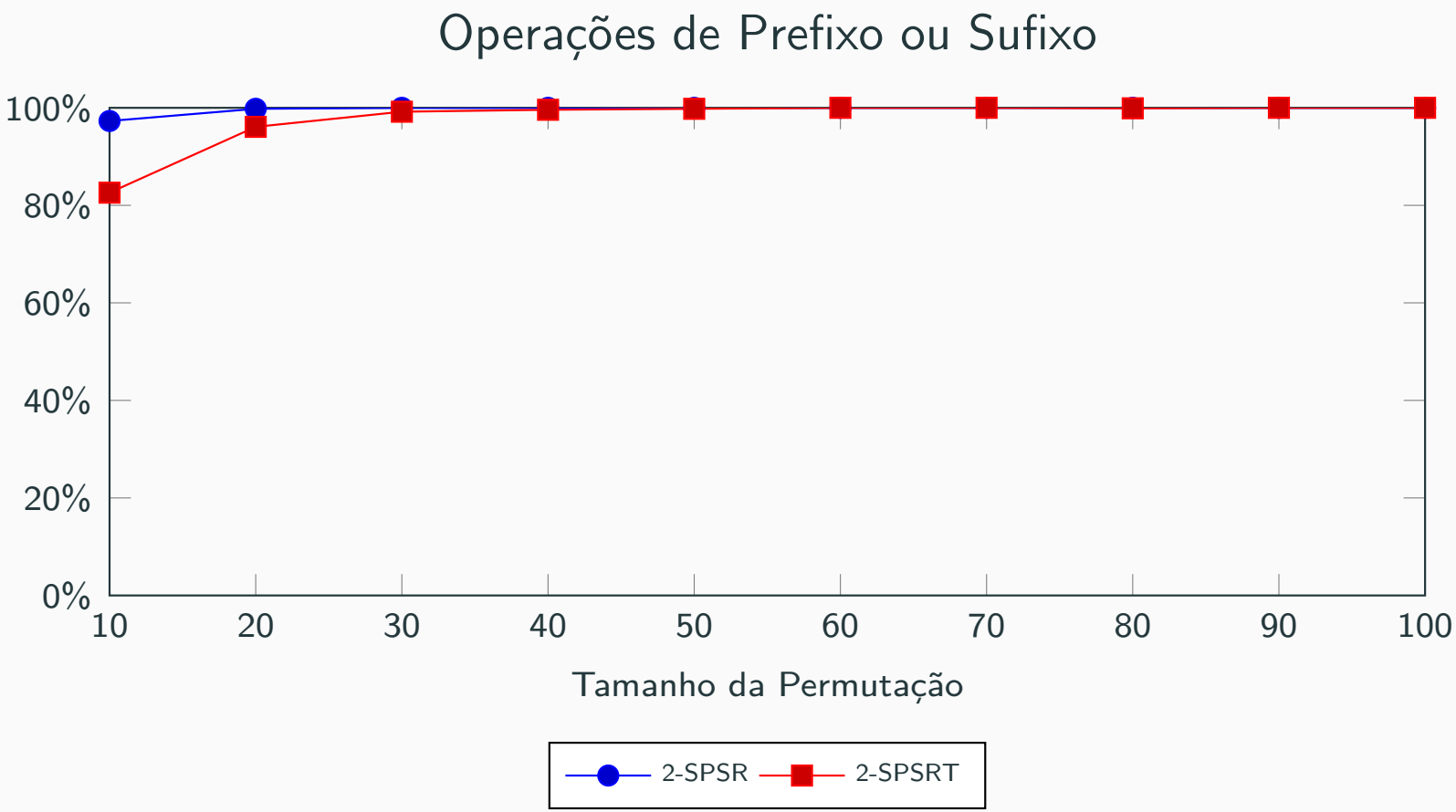


Figura 11: Porcentagem de soluções melhoradas utilizando Look Ahead.

Iterative Sliding Window

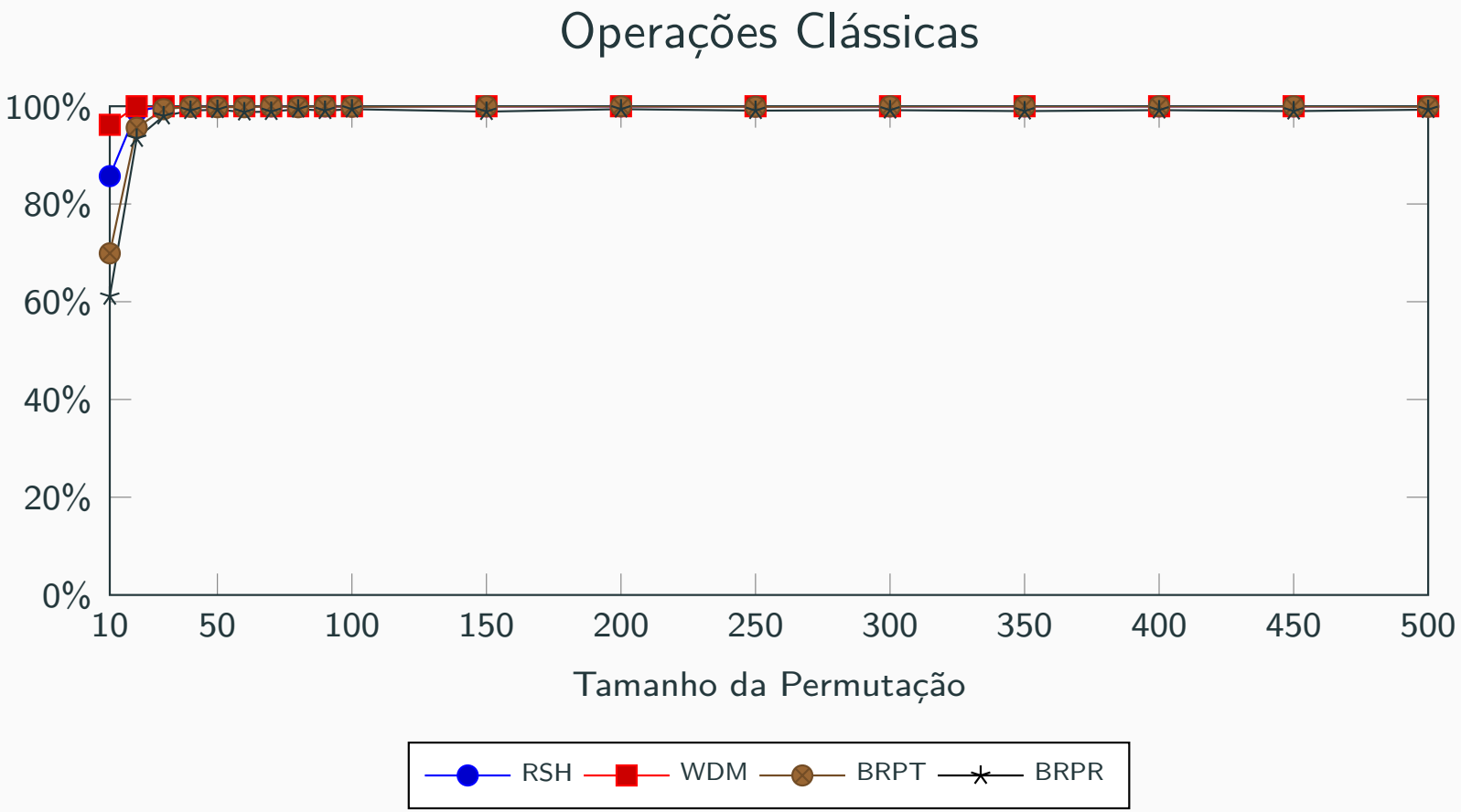


Figura 12: Porcentagem de soluções melhoradas utilizando Iterative Sliding Window.

Iterative Sliding Window

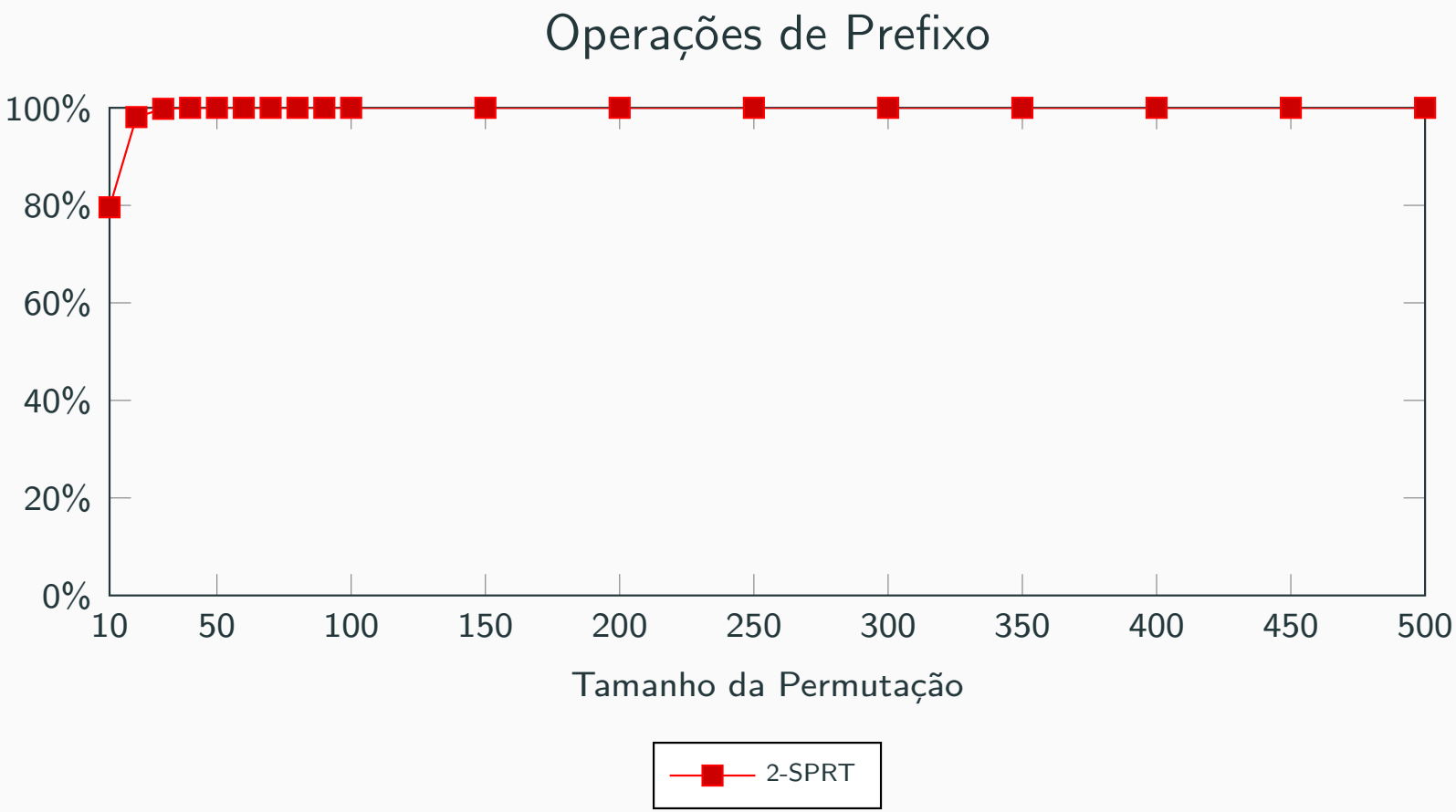


Figura 13: Porcentagem de soluções melhoradas utilizando Iterative Sliding Window.

Iterative Sliding Window

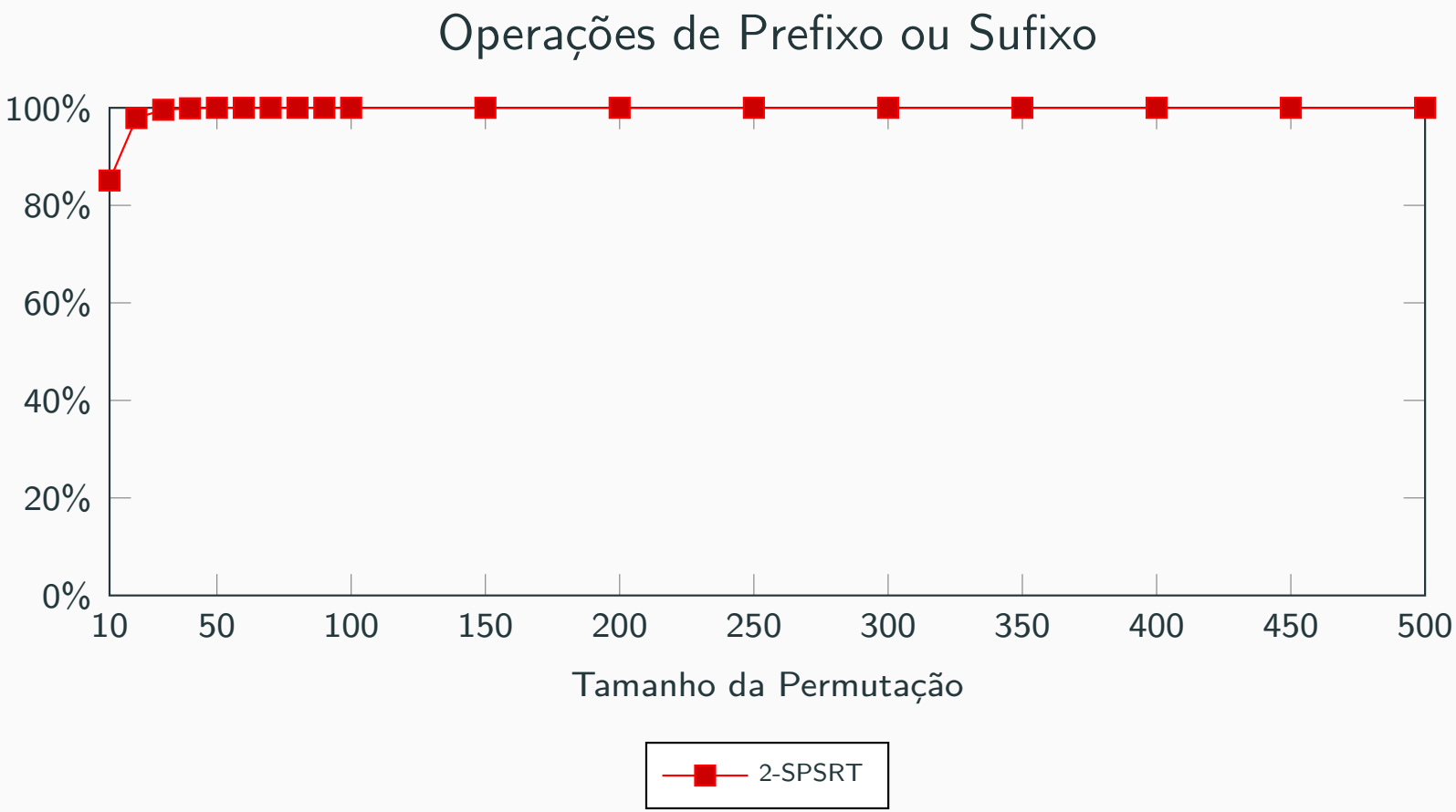


Figura 14: Porcentagem de soluções melhoradas utilizando Iterative Sliding Window.

Look Ahead + Sliding Window

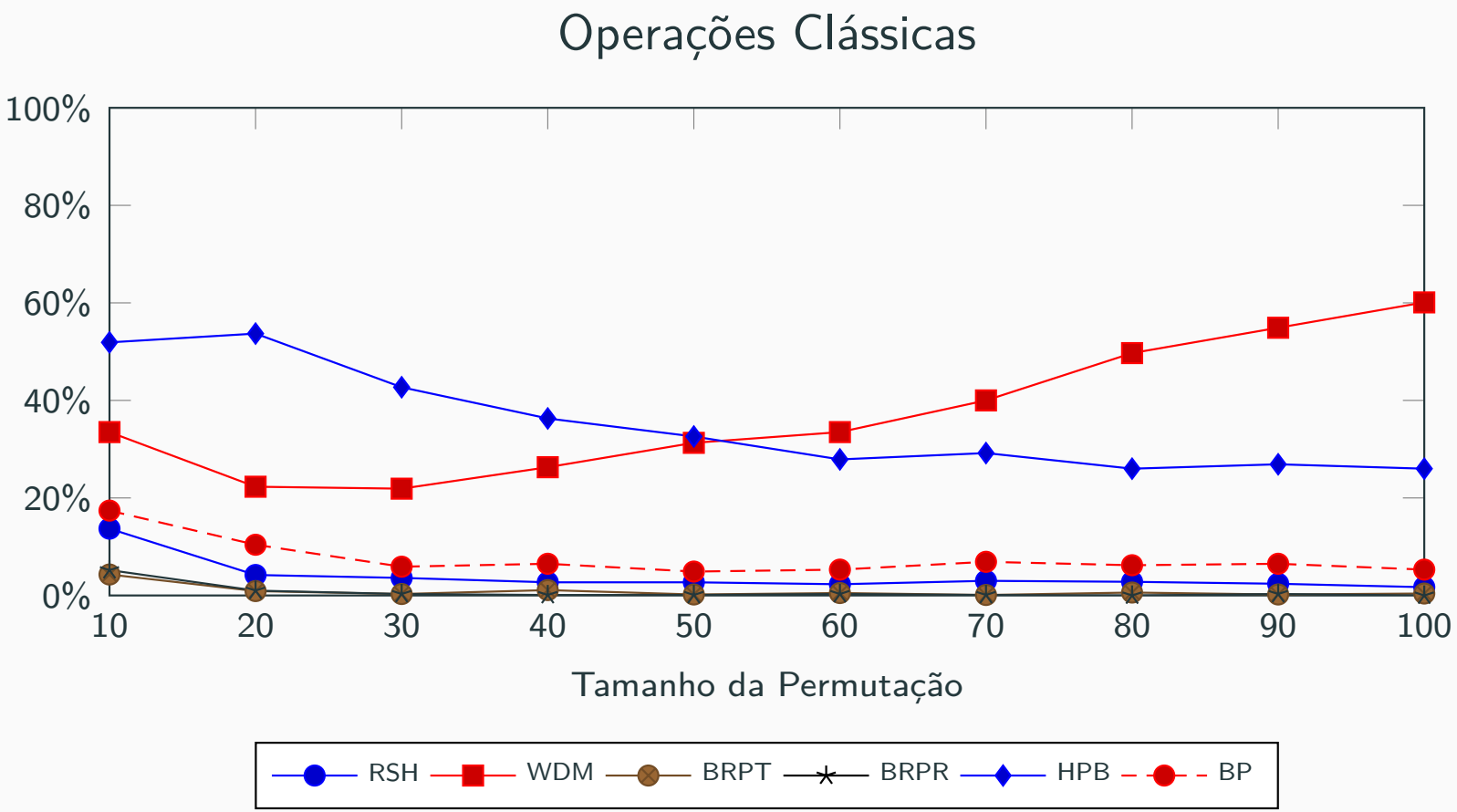


Figura 15: Porcentagem de soluções melhoradas utilizando Look Ahead + Sliding Window.

Look Ahead + Sliding Window

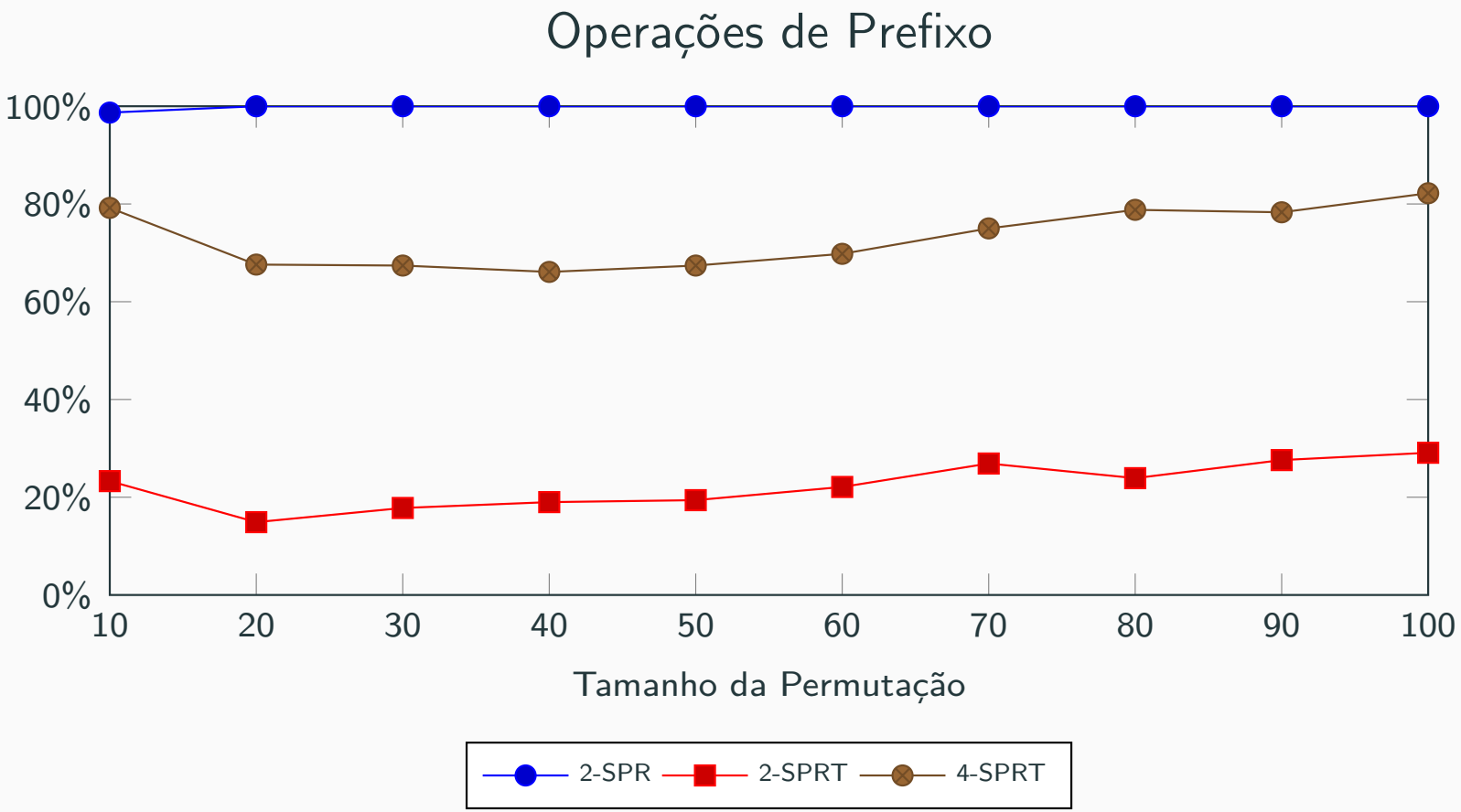


Figura 16: Porcentagem de soluções melhoradas utilizando Look Ahead + Sliding Window.

Look Ahead + Sliding Window

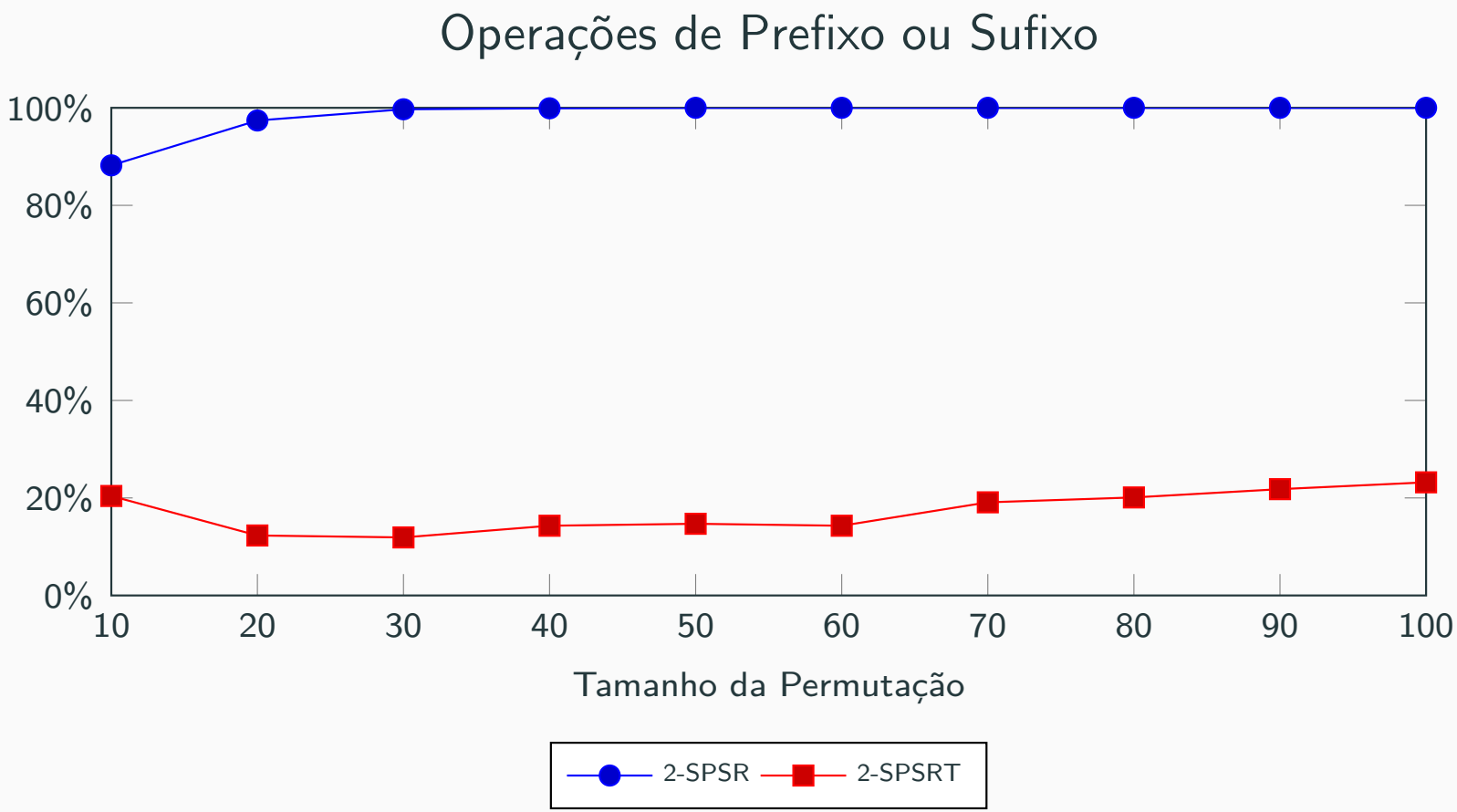


Figura 17: Porcentagem de soluções melhoradas utilizando Look Ahead + Sliding Window.

Algoritmos de Aproximação

Ponderação Adotada

- Problema de Ordenação de Permutações com Sinais por Reversões e Transposições Ponderadas.
- Ponderação utilizada:
  - Reversões $w_\rho = 2$.
  - Transposições $w_\tau = 3$.
- Bader e coautores [2] acreditam que essa razão de peso é realista para a maioria dos conjuntos de dados biológicos.
- Estudo realizado por Eriksen [4], que otimizou os pesos utilizados para cada evento.

Algoritmos Desenvolvidos

Algoritmos desenvolvidos para o problema de Ordenação de Permutações com Sinais por Reversões e Transposições Ponderadas:

Breakpoints

- 3-aproximado.
- 2-aproximado.

Grafo de Ciclos

- 5/3-aproximado.
- 3/2-aproximado.

Conjunto de Dados

Analisar o comportamento dos algoritmos em diferentes cenários.

Organização do nosso conjuntos de dados:

- Dividido em 5 grupos com diferentes características.
- Cada grupo com diversos conjuntos de permutações.
- Cada conjunto de permutações tem um total de 10.000 permutações de um tamanho específico.

Resultados

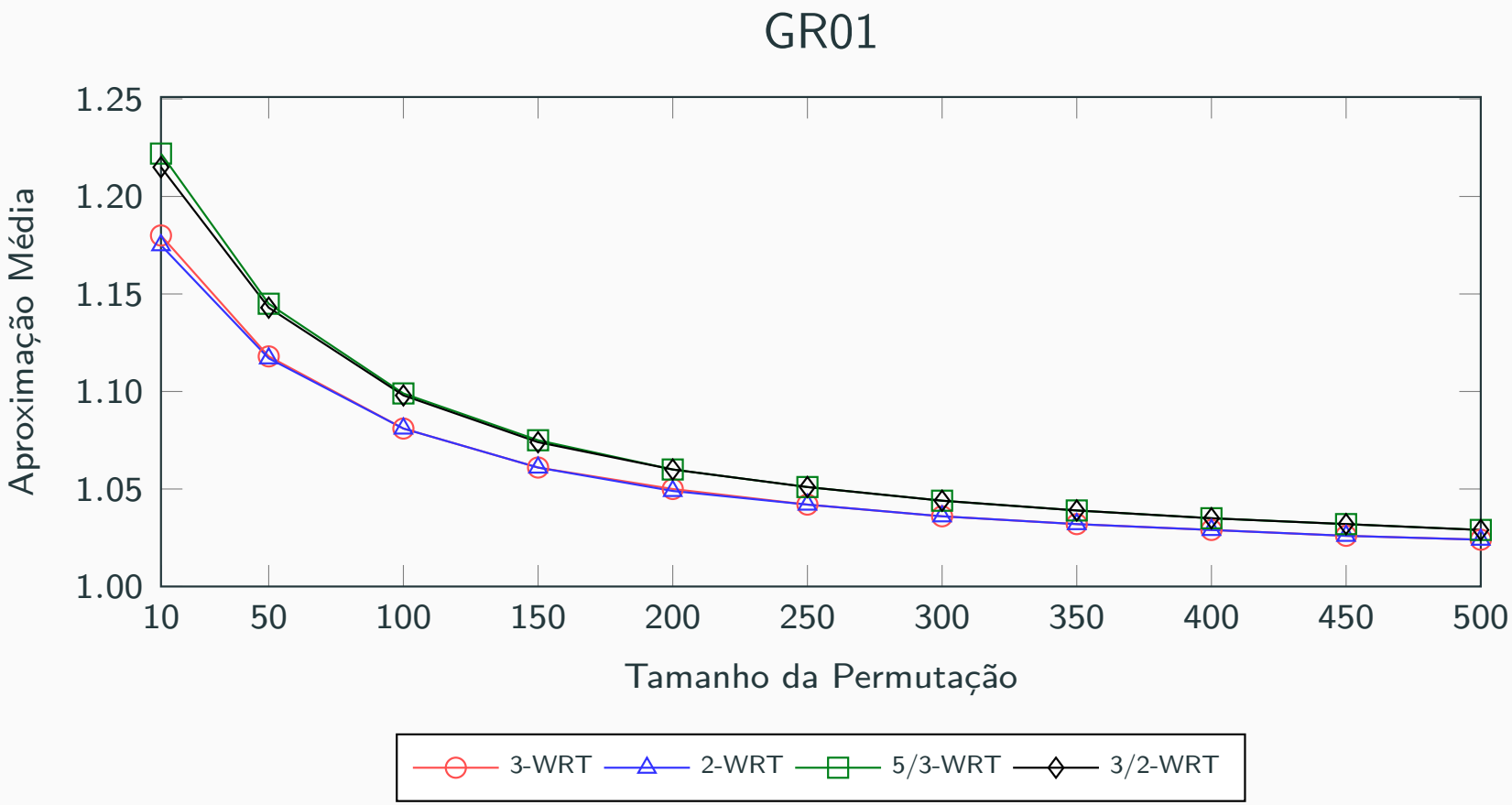


Figura 18: Permutações com o maior número possível de breakpoints.

Resultados

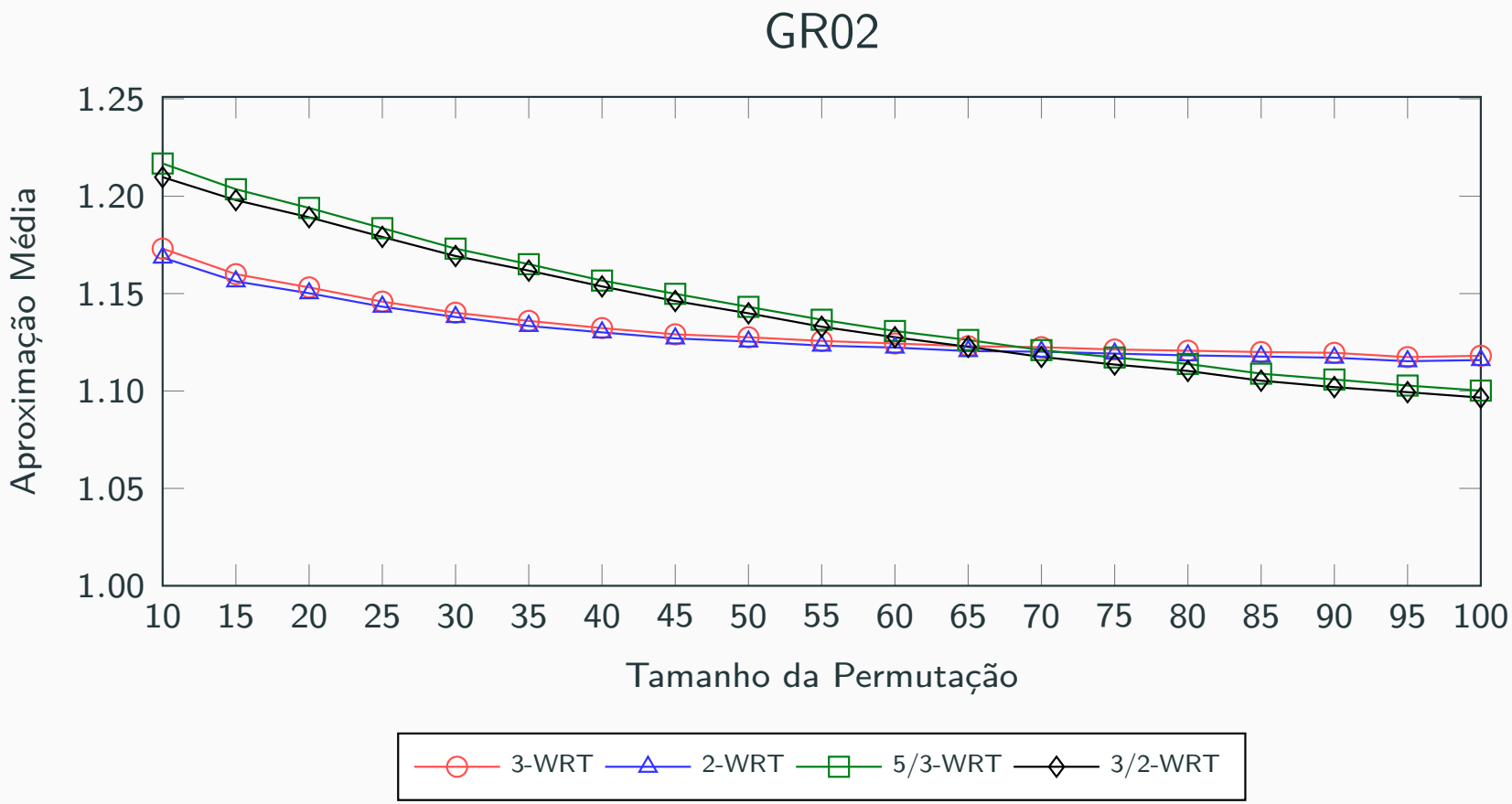


Figura 19: Aplicando uma quantidade fixa de 20 operações, sendo 10 reversões e 10 transposições.

Resultados

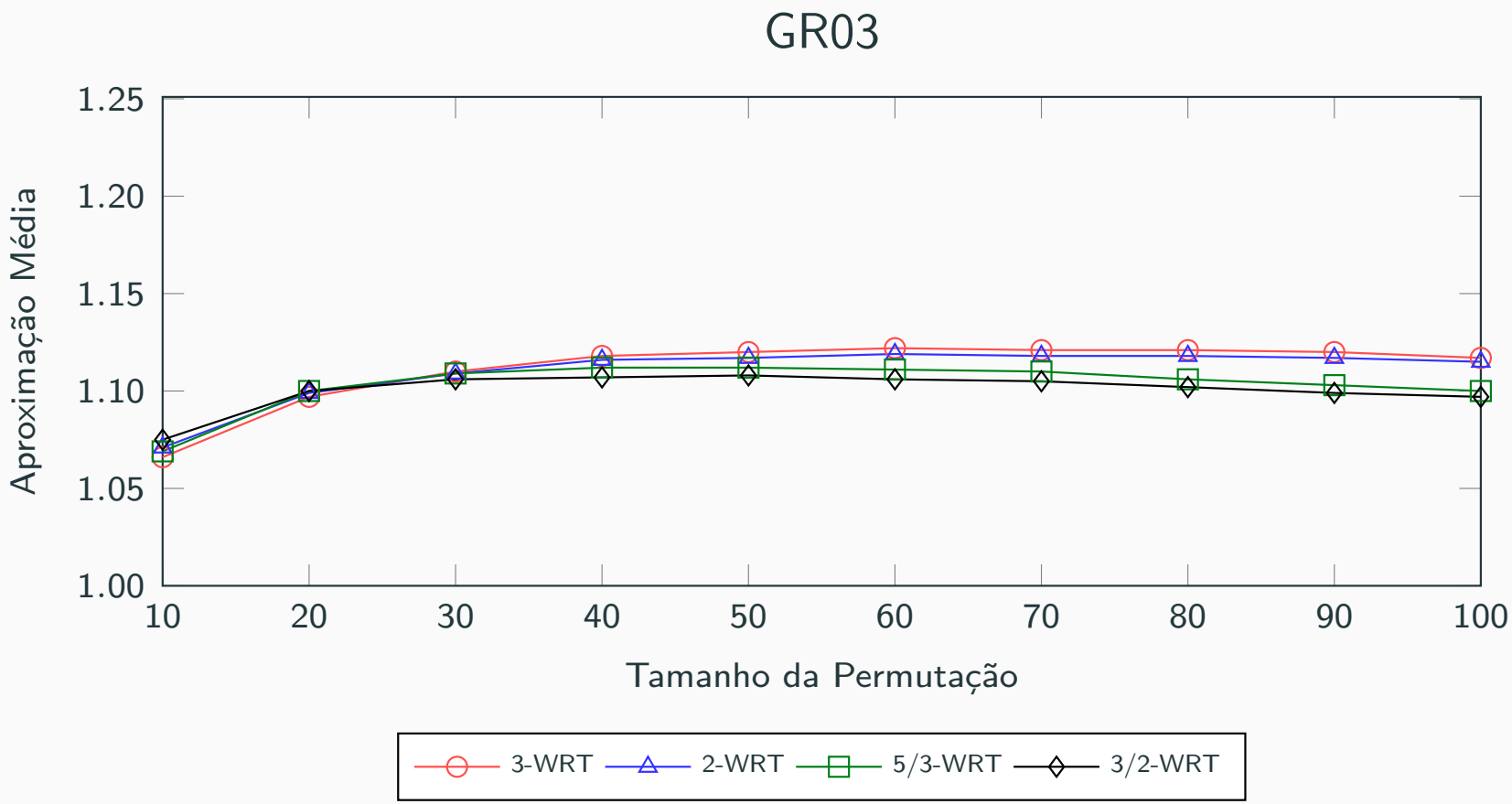


Figura 20: Aplicando 20% do tamanho da permutação em operações, sendo 10% reversões e 10% transposições.

Resultados

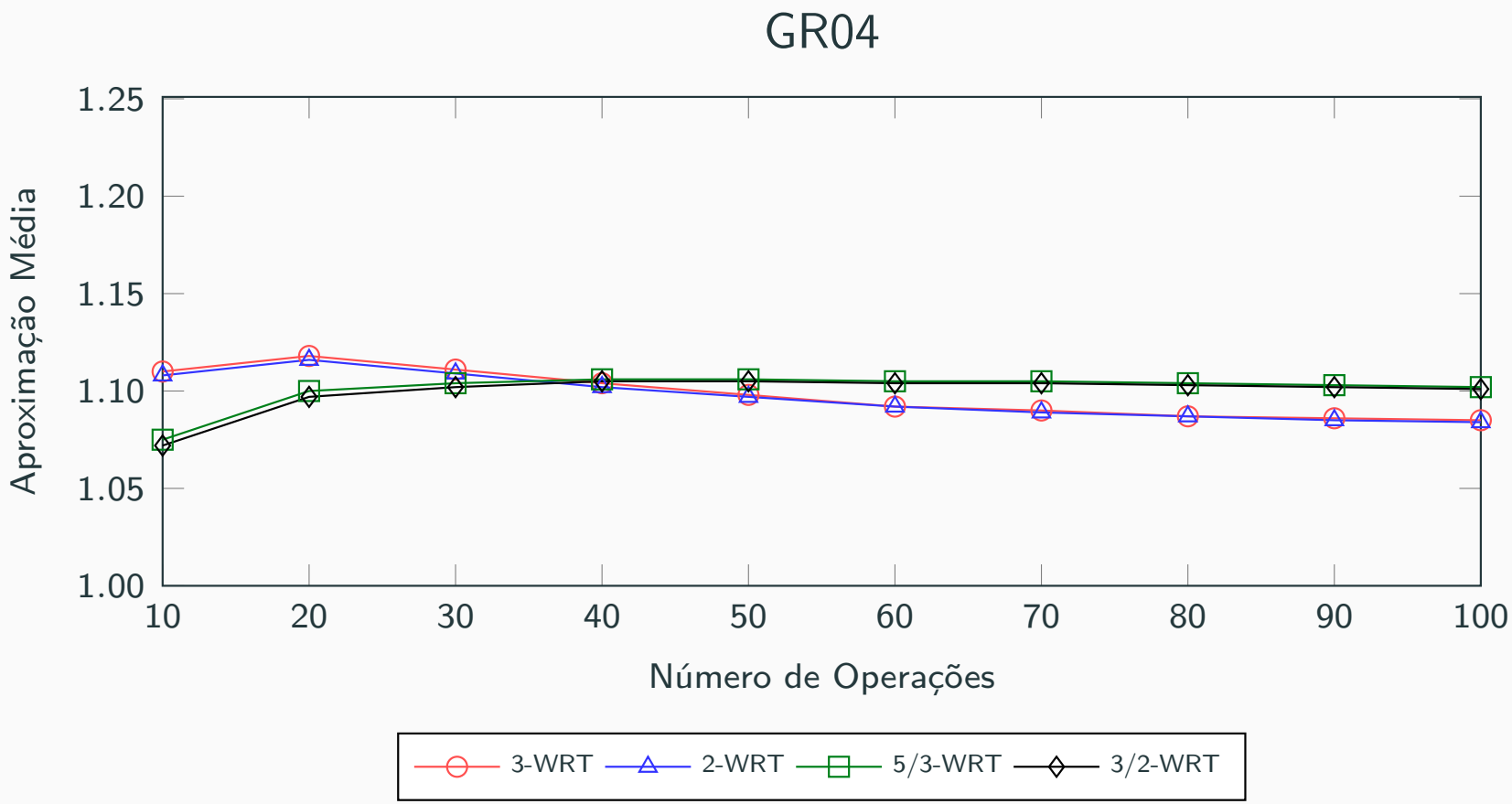


Figura 21: Permutações de tamanho 100 aplicando um número variável de operações, sendo 50% reversões e 50% transposições.

Resultados

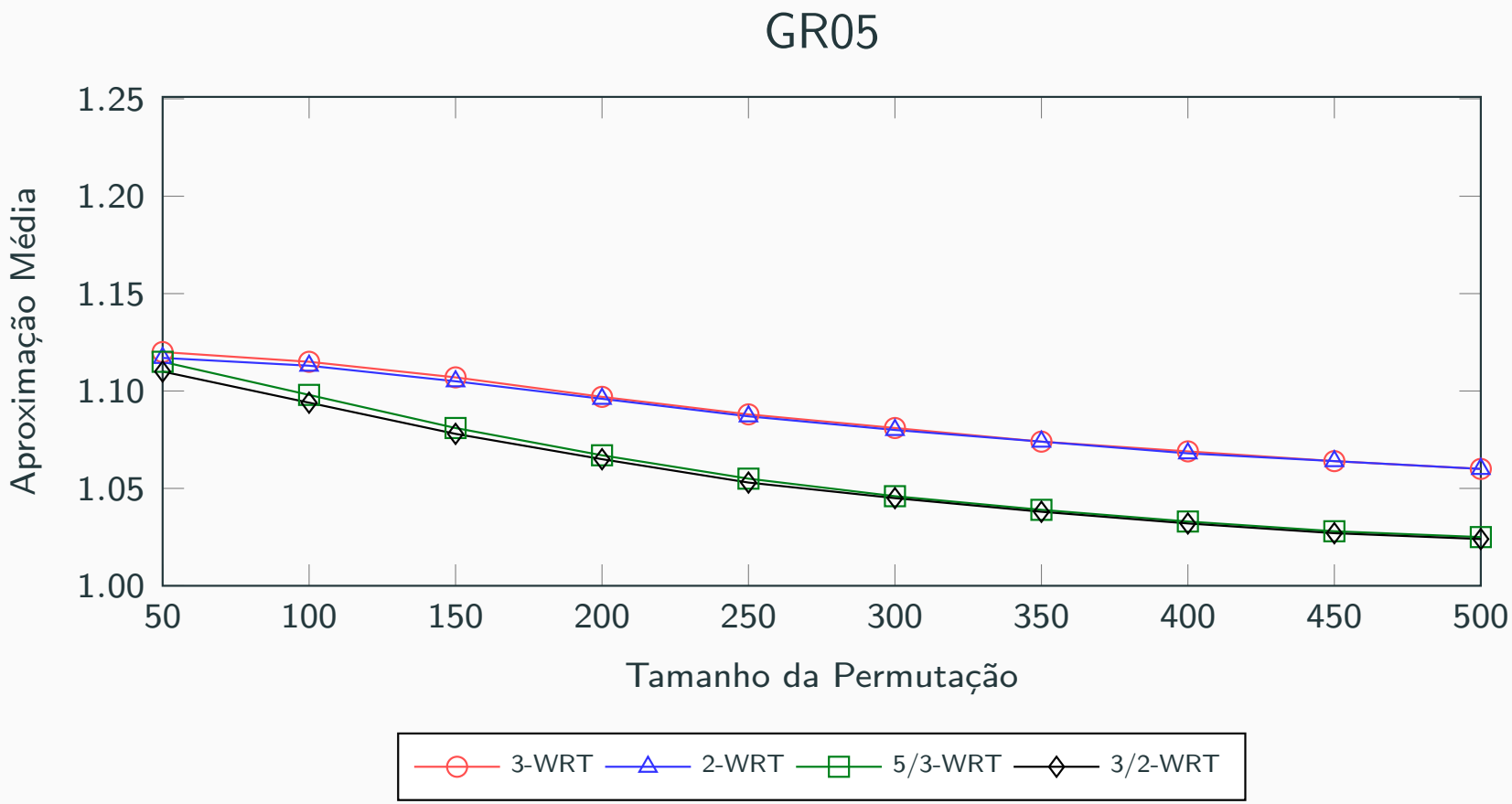


Figura 22: Número de operações igual a 20% do tamanho da permutação, sendo 60% reversões e 40% transposições.

Investigando Diferentes Ponderações

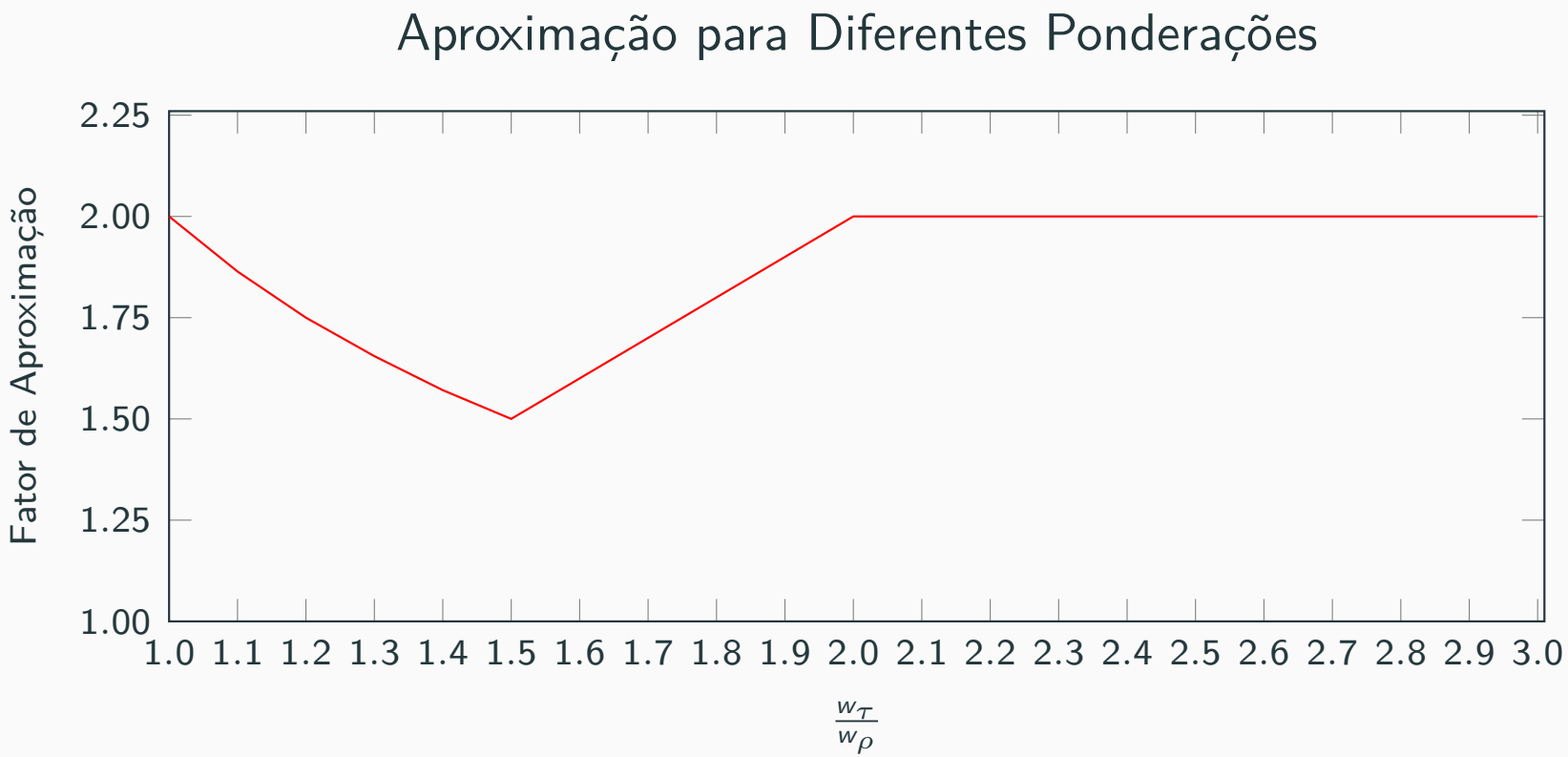


Figura 23: Fatores de aproximação adotando diferentes pesos para w_τ e w_ρ.

Conclusões

Conclusões

Problema de Ordenação de Permutações com Sinais por Reversões e Transposições:

- Três heurísticas para melhorar soluções de algoritmos.
- Operações clássicas, prefixo e prefixo ou sufixo.
- Uso conjunto de duas heurísticas para obter resultados melhores.
- Aplicamos nossas heurísticas em algoritmos da literatura.

Problema de Ordenação de Permutações com Sinais por Reversões e Transposições Poderadas:

- Quatro algoritmos de aproximação considerando pesos 2 e 3 para reversões e transposições, respectivamente.
- Realizamos experimentos com vários conjuntos de permutações com características distintas.
- Análise considerando relações distintas de peso entre as operações de transposição e reversão.

Contribuições

Heurísticas Sliding Window e Look Ahead

K. L. Brito, A. R. Oliveira, U. Dias and Z. Dias. Heuristics for the Sorting Signed Permutations by Reversals and Transpositions Problem. In Proceedings of the 5th International Conference on Algorithms for Computational Biology (AlCoB 2018), Hong Kong, China, 2018

Referências


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, 8:483–491, 2001.


M. Bader, M. I. Abouelhoda, and E. Ohlebusch.
A Fast Algorithm for the Multiple Genome Rearrangement Problem with Weighted Reversals and Transpositions.
BMC Bioinformatics, 9(1):1–13, 2008.


V. Bafna and P. A. Pevzner.
Sorting by Transpositions.
SIAM Journal on Discrete Mathematics, 11(2):224–240, 1998.


N. Eriksen.
Combinatorics of genome rearrangements and phylogeny, 2001.

Referências


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, 1999.

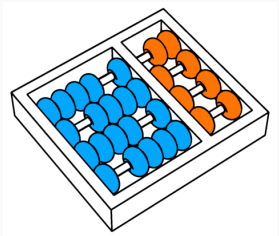

C. N. Lintzmayer, G. Fertin, and Z. Dias.
Sorting Permutations by Prefix and Suffix Rearrangements.
Journal of Bioinformatics and Computational Biology, 15(1):1750002, 2017.


A. Rahman, S. Shatabda, and M. Hasan.
An Approximation Algorithm for Sorting by Reversals and Transpositions.
Journal of Discrete Algorithms, 6(3):449–457, 2008.

Referências


M. E. M. T. Walter, Z. Dias, and J. 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.

Obrigado!



Ordenação de Permutações com Sinais por Reversões e Transposições

Klairton de Lima Brito

Orientador: Zanoni Dias

Instituto de Computação
Universidade Estadual de Campinas
Dissertação de Mestrado