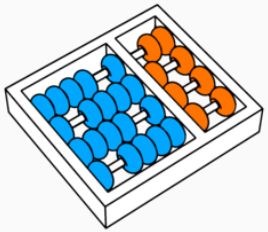


UNICAMP

Heurísticas para Problemas de Rearranjo de Genomas com Genes Multiplicados

Aluno: Gabriel Henriques Siqueira
Orientador: Prof. Dr. Zanoni Dias
Coorientador: Dr. André Rodrigues Oliveira

Instituto de Computação – Unicamp

Roteiro

Motivação

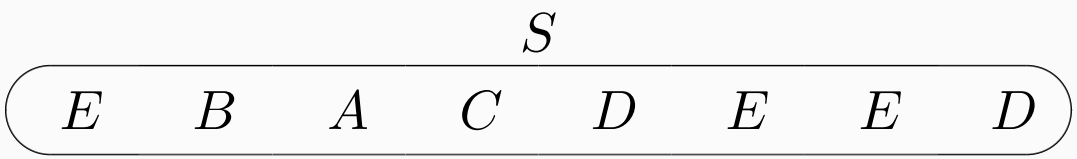
Motivação

- Determinar a distância evolutiva entre genomas considerando eventos conservativos que afetam grandes porções do genoma, chamados eventos de rearranjo conservativos.
- Geralmente, assume-se que não existem repetições de genes.
- Propomos heurísticas para o caso em que os genes podem estar multiplicados.

Conceitos

Representação por Strings

- Representamos um genoma por uma string S.
- Cada caractere corresponde a um gene.
- Algumas notações:
  - $|S|$ = número de caracteres em S.
  - S_i = i-ésimo caractere de S.
  - Σ_S = conjunto de caracteres de S (rótulos).
  - $dup(S)$ = conjunto de rótulos duplicados de S.
  - $multi(S)$ = conjunto de rótulos multiplicados de S.
  - $occ(\alpha, S)$ = ocorrência de um rótulo α na string S.



$|S| = 8$
$S_1 = E, \quad S_3 = A$
$\Sigma_S = \{A, B, C, D, E\}$
$dup(S) = \{D\}$
$multi(S) = \{D, E\}$
$occ(E, S) = 3$

Representação por Strings

- Quando conhecida, a orientação dos genes é representada por um sinal $+$ ou $-$.
- Σ_S, $dup(S)$, $multi(S)$ e $occ(\alpha, S)$ desconsideram a orientação.

$$\overbrace{\underbrace{+E \quad -B \quad -A \quad +C \quad +D \quad +E \quad +E \quad -D}}^{S}$$

$|S| = 8$
$S_1 = +E, \quad S_3 = -A$
$\Sigma_S = \{A, B, C, D, E\}$
$dup(S) = \{D\}$
$multi(S) = \{D, E\}$
$occ(E, S) = 3$

Representação por Strings

- S e P são balanceadas se:
  - $\Sigma_S = \Sigma_P$;
  - $occ(\alpha, S) = occ(\alpha, P), \forall \alpha \in \Sigma_S$.
- Lidamos apenas com strings balanceadas.

S

E	B	A	C	D	E	D

$occ(A, S) = 1,\ occ(E, S) = 2$

P

D	D	A	B	E	E	C

$occ(A, P) = 1,\ occ(E, P) = 2$

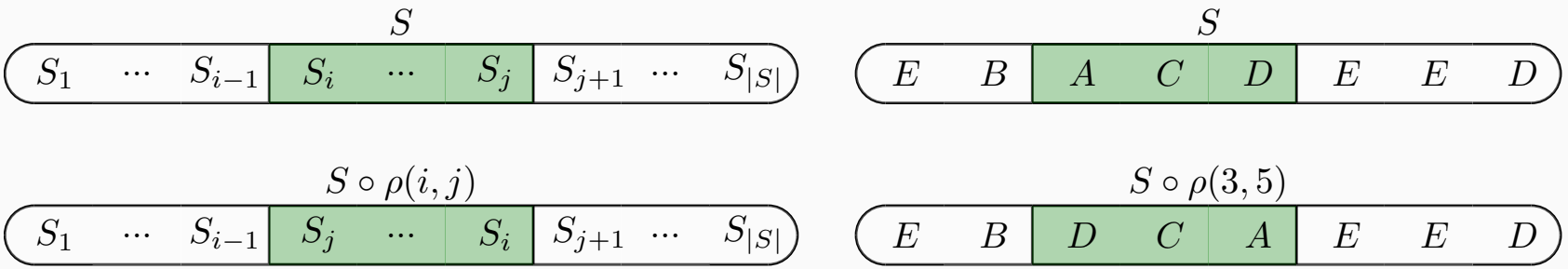
Q

E	A	A	C	D	D	B

$occ(A, Q) = 2,\ occ(E, Q) = 1$

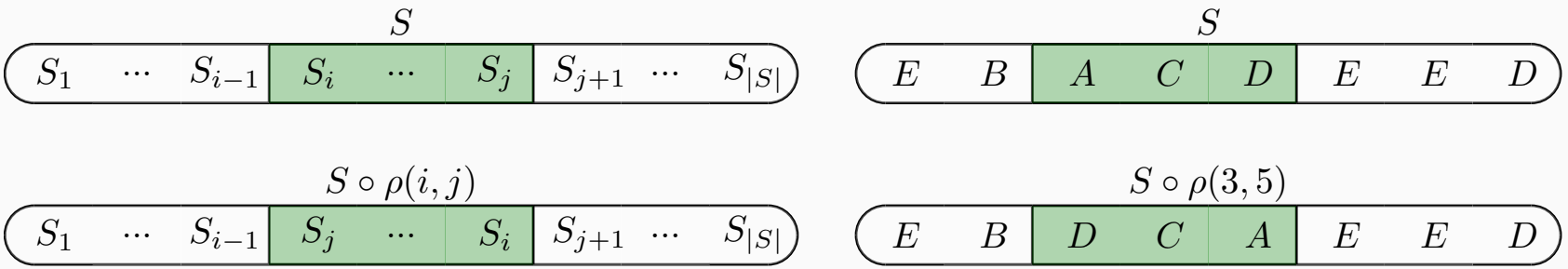
Eventos de Rearranjo

- Reversão ($\rho(i, j)$):

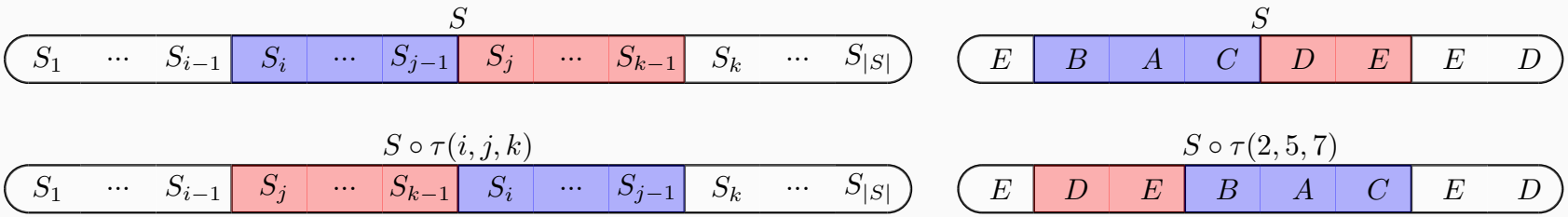


Eventos de Rearranjo

- Reversão ($\rho(i,j)$):

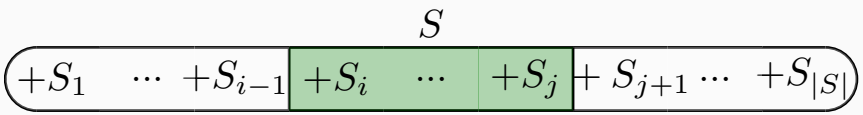


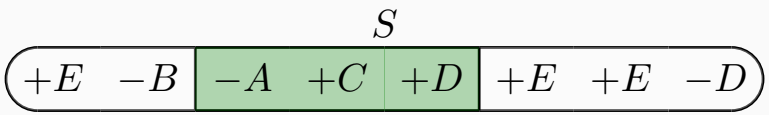
- Transposição ($\tau(i,j,k)$):

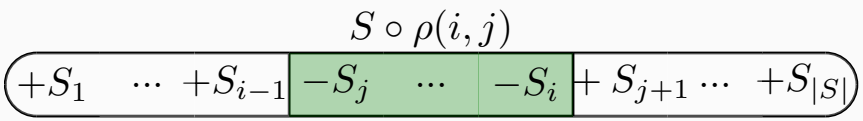


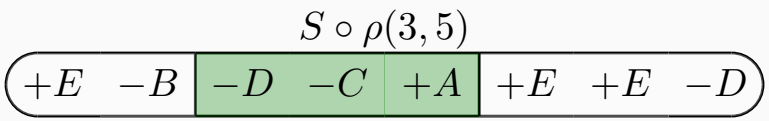
Eventos de Rearranjo

- Reversão ($\rho(i,j)$):

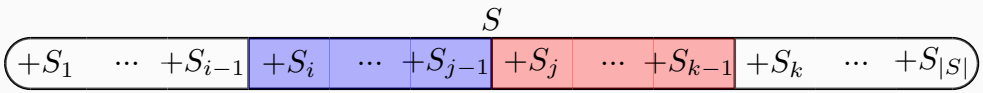


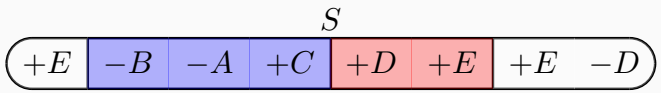


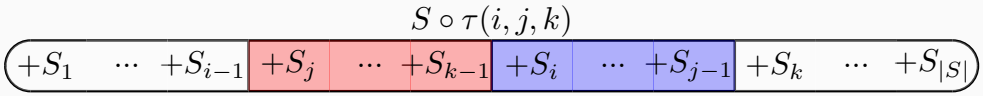


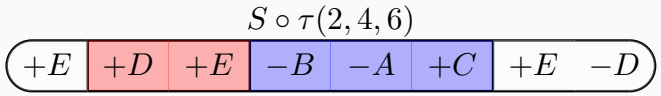


- Transposição ($\tau(i,j,k)$):









Eventos de Rearranjo

- Modelo de rearranjo ($\mathcal{M}$): conjunto de eventos de rearranjo permitidos.
- Distância de rearranjo ($d_{\mathcal{M}}(S, P)$): menor número de operações correspondentes a rearranjos do modelo $\mathcal{M}$ necessárias para transformar S em P.

Objetivos

O objetivo é estudar os seguintes problemas com genes duplicados ou multiplicados:

- Distância de Reversão em Strings sem Sinais (DR).
- Distância de Reversão em Strings com Sinais ($D\bar{R}$).
- Distância de Transposição em Strings sem Sinais (DT).
- Distância de Reversão e Transposição em Strings sem Sinais (DRT).
- Distância de Reversão e Transposição em Strings com Sinais ($D\bar{R}T$).

Resultados Conhecidos

Distância de Reversão em strings sem sinais:

- NP-Difícil (Caprara 1999).
- Aproximação com fator 1.375 sem caracteres repetidos (Berman et al. 2002).
- Aproximação com fator 2.2074 com caracteres duplicados (Goldstein et al. 2005).
- Aproximação com fator $\theta(k)$ para o caso geral, onde k é o número máximo de cópias de um caractere (Kolman e Waleń 2007b).
- Aproximação com fator $\log n \log^* n$ para o caso geral, onde n é o tamanho das strings (Cormode e Muthukrishnan 2007).

Resultados Conhecidos

Distância de Reversão em strings com sinais:

- Polinomial sem caracteres repetidos (Hannenhalli e Pevzner 1999).
- NP-Difícil com caracteres duplicados ou repetidos (Chen et al. 2005; Radcliffe et al. 2005).
- Aproximação com fator 2.2074 com caracteres duplicados (Goldstein et al. 2005).
- Aproximação com fator $\theta(k)$ para o caso geral (Kolman e Waleń 2007b).
- Aproximação com fator $\log n \log^* n$ para o caso geral (Cormode e Muthukrishnan 2007).

Resultados Conhecidos

Distância de Transposição em strings sem sinais:

- NP-Difícil (Bulteau et al. 2012).
- Aproximação com fator 1.375 sem caracteres repetidos (Berman et al. 2002).
- Aproximação com fator 3.311 com caracteres duplicados (Goldstein et al. 2005).
- Aproximação com fator $\theta(k)$ para o caso geral (Kolman e Waleń 2007b).
- Aproximação com fator $\log n \log^* n$ para o caso geral (Cormode e Muthukrishnan 2007).

Resultados Conhecidos

Distância de Reversão e Transposição em strings sem sinais:

- NP-Difícil (Oliveira et al. 2019).
- Aproximação com fator 2α sem caracteres repetidos, onde α é a aproximação do algoritmo usado para decomposição em ciclos (Rahman et al. 2008). O melhor valor conhecido para α é $1.4167 + \epsilon$ (Chen 2010).
- Aproximação com fator 3.311 com caracteres duplicados (Goldstein et al. 2005).
- Aproximação com fator $\theta(k)$ para o caso geral (Kolman e Waleń 2007b).
- Aproximação com fator $\log n \log^* n$ para o caso geral (Cormode e Muthukrishnan 2007).

Resultados Conhecidos

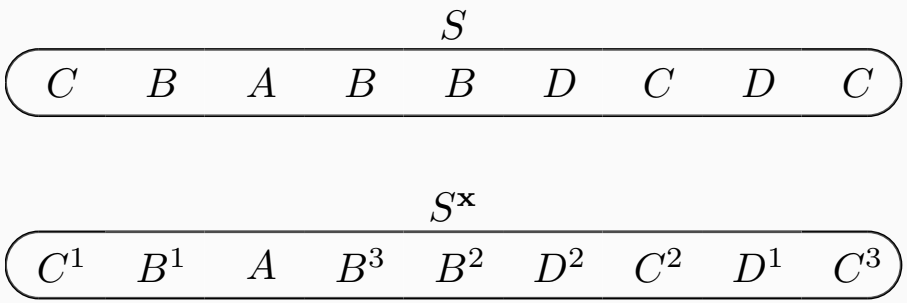
Distância de Reversão e Transposição em strings com sinais:

- NP-Difícil (Oliveira et al. 2019).
- Aproximação com fator 2 sem caracteres repetidos (Walter et al. 1998).
- Aproximação com fator 3.311 com caracteres duplicados (Goldstein et al. 2005).
- Aproximação com fator $\theta(k)$ para o caso geral (Kolman e Waleń 2007b).
- Aproximação com fator $\log n \log^* n$ para o caso geral (Cormode e Muthukrishnan 2007).

Heurísticas de Mapeamentos

Mapeamentos em Permutações

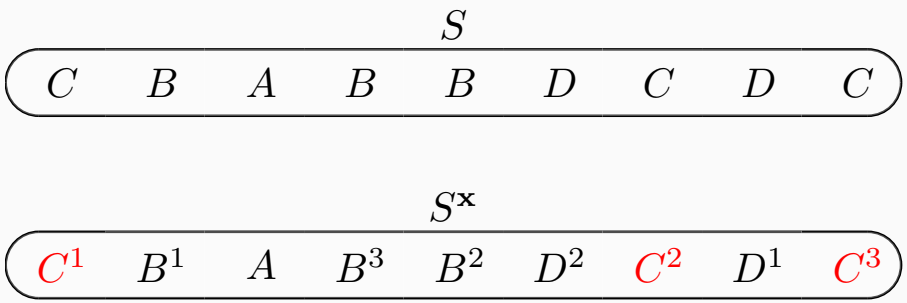
- Mapeamos as strings em permutações (strings sem caracteres repetidos), assim, podemos usar resultados já conhecidos para permutações.
- Utilizamos um vetor de permutações para representar um mapeamento.
- Para obter um vizinho, trocamos uma permutação pela próxima na ordem lexicográfica.



$$\mathbf{x} = \overset{\text{B}}{\boxed{1\ 3\ 2}}\overset{\text{C}}{\boxed{1\ 2\ 3}}\overset{\text{D}}{\boxed{2\ 1}}$$

Mapeamentos em Permutações

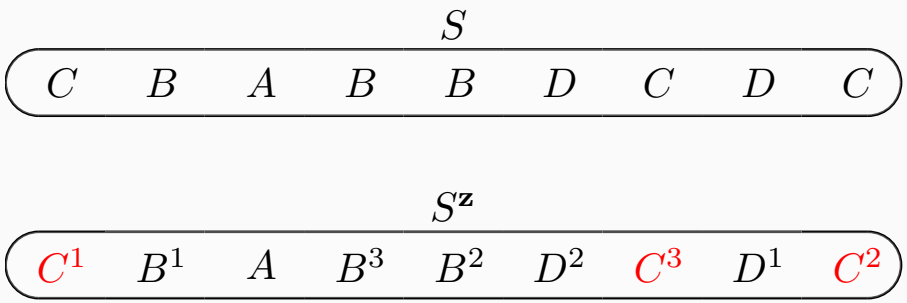
- Mapeamos as strings em permutações (strings sem caracteres repetidos), assim, podemos usar resultados já conhecidos para permutações.
- Utilizamos um vetor de permutações para representar um mapeamento.
- Para obter um vizinho, trocamos uma permutação pela próxima na ordem lexicográfica.

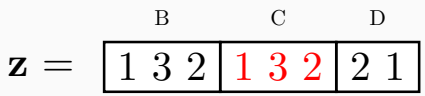


$$\mathbf{x} = \begin{array}{|c|c|c|}\hline \overset{\text{B}}{1\ 3\ 2} & \overset{\text{C}}{1\ 2\ 3} & \overset{\text{D}}{2\ 1} \\ \hline \end{array}$$

Mapeamentos em Permutações

- Mapeamos as strings em permutações (strings sem caracteres repetidos), assim, podemos usar resultados já conhecidos para permutações.
- Utilizamos um vetor de permutações para representar um mapeamento.
- Para obter um vizinho, trocamos uma permutação pela próxima na ordem lexicográfica.





Heurísticas Utilizando Mapeamentos

Heurísticas Utilizando Mapeamentos

S

$C\ B\ A\ B\ D\ C\ D\ B$

$S^{\mathbf{x_2}}$

$C^1\ B^1\ A\ B^2\ D^1\ C^2\ D^2\ B^3$

$S^{\mathbf{x_1}}$

$C^2\ B^1\ A\ B^3\ D^1\ C^1\ D^2\ B^2$

$S^{\mathbf{x_3}}$

$C^1\ B^3\ A\ B^2\ D^2\ C^2\ D^1\ B^1$

$A\ C^1\ B^1\ D^1\ B^2\ B^3\ D^2\ C^2$

$P^{\mathbf{y}}$

$A\ C\ B\ D\ B\ B\ D\ C$

P

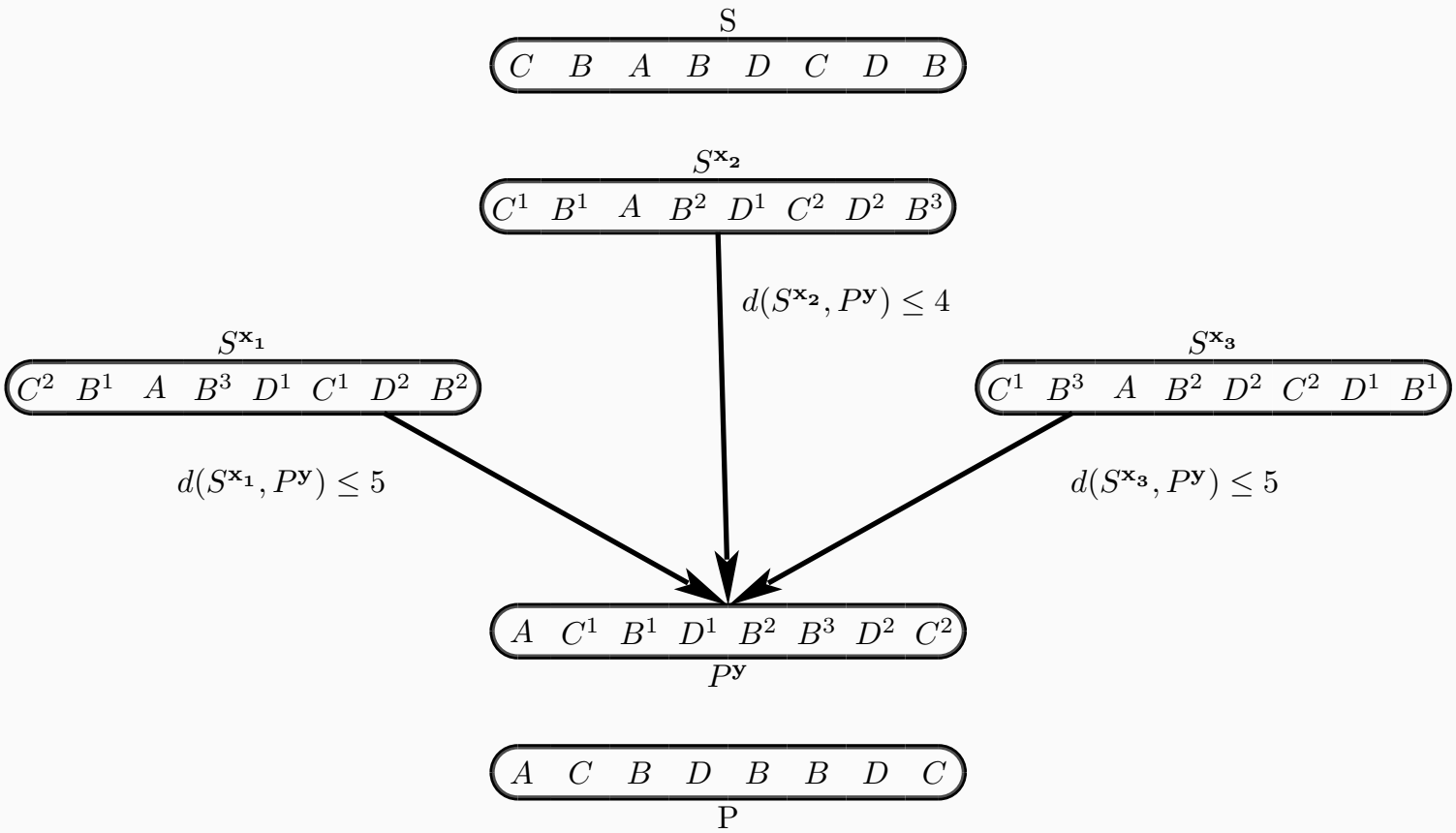
$\mathbf{y} =$ B C D: 1 2 3 | 1 2 | 1 2

$\mathbf{x_1} =$ B C D: 1 3 2 | 2 1 | 1 2

$\mathbf{x_2} =$ B C D: 1 2 3 | 1 2 | 1 2

$\mathbf{x_3} =$ B C D: 3 2 1 | 1 2 | 2 1

Heurísticas Utilizando Mapeamentos



$\mathbf{y} =$

B	C	D
1 2 3	1 2	1 2

$\mathbf{x_1} =$

B	C	D
1 3 2	2 1	1 2

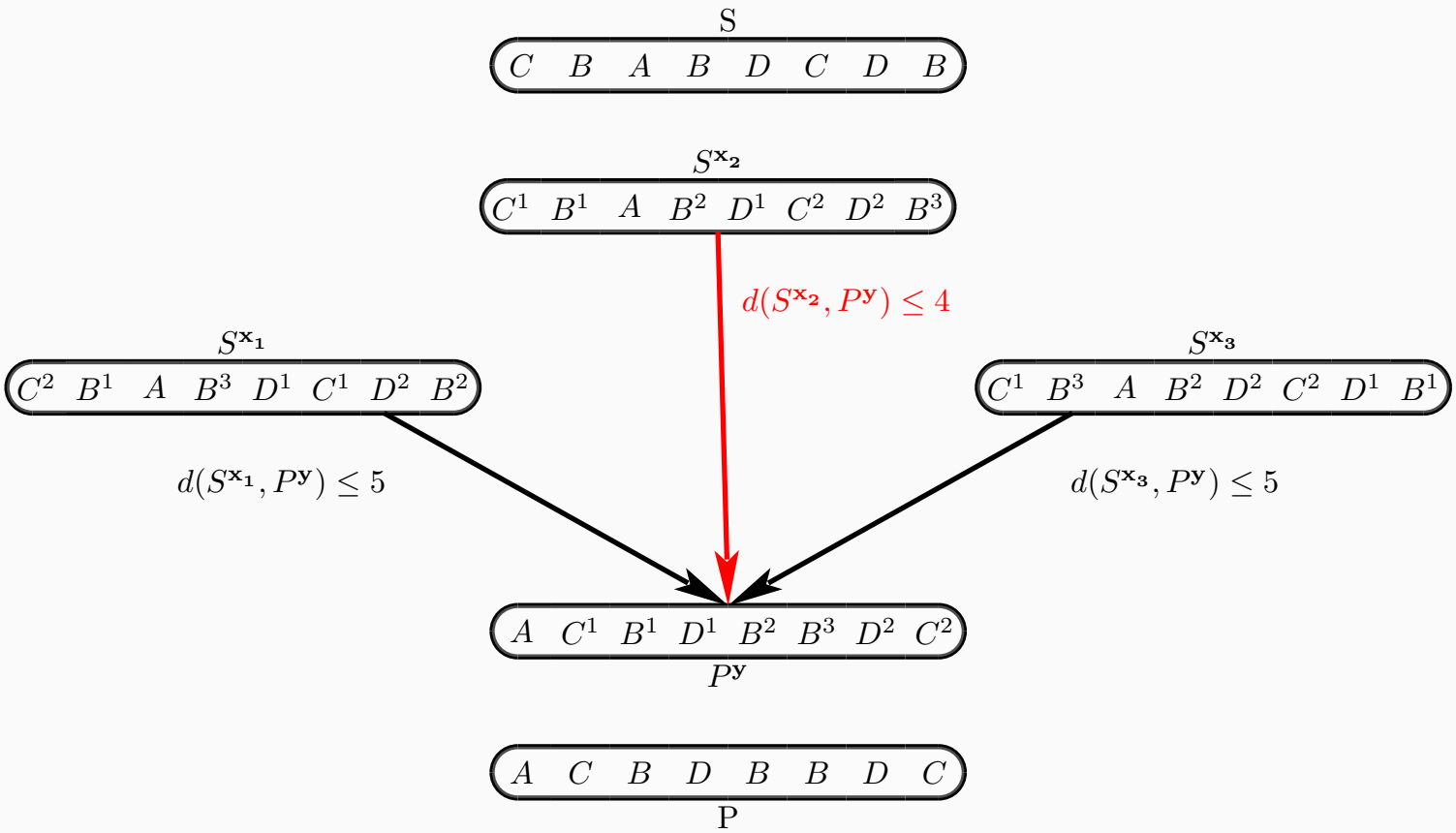
$\mathbf{x_2} =$

B	C	D
1 2 3	1 2	1 2

$\mathbf{x_3} =$

B	C	D
3 2 1	1 2	2 1

Heurísticas Utilizando Mapeamentos



$\mathbf{y} =$ B C D: 1 2 3 | 1 2 | 1 2 $\quad \mathbf{x_1} =$ B C D: 1 3 2 | 2 1 | 1 2 $\quad \mathbf{x_2} =$ B C D: 1 2 3 | 1 2 | 1 2 $\quad \mathbf{x_3} =$ B C D: 3 2 1 | 1 2 | 2 1

Heurísticas Utilizando Mapeamentos

- Mapeamentos Aleatórios (MA):
  - Cada elemento dos mapeamentos é escolhido de forma aleatória e uniforme.

Heurísticas Utilizando Mapeamentos

- Mapeamentos Aleatórios (MA):
  - Cada elemento dos mapeamentos é escolhido de forma aleatória e uniforme.
- Busca Local (BL):
  - Utiliza um conjunto inicial de mapeamentos aleatórios.
  - Explora os vizinhos do melhor mapeamento conhecido até o momento.
  - Não explora um mapeamento mais de uma vez.

Heurísticas Utilizando Mapeamentos

- GRASP:
  - Utiliza um conjunto inicial de mapeamentos aleatórios.
  - Etapa de construção:

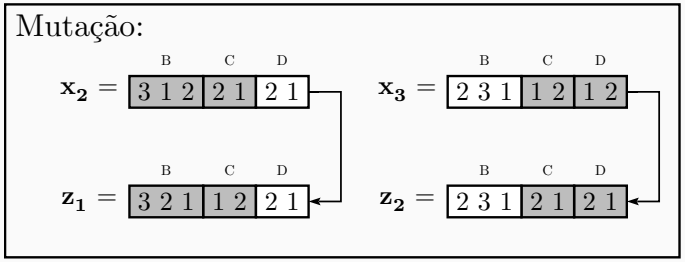
$$prob(\text{RCL}, \alpha, \phi) = \frac{freq(\text{RCL}, \alpha, \phi)}{\sum_{\xi \in Sym(\alpha, S)} freq(\text{RCL}, \alpha, \xi)} = \frac{freq(\text{RCL}, \alpha, \phi)}{|\text{RCL}|}$$

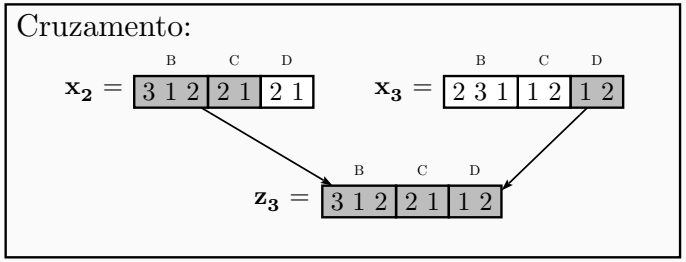
RCL	
$\mathbf{x_1} =$ <table><tr><th></th><th>B</th><th>C</th><th>D</th></tr><tr><td></td><td>3 1 2</td><td>1 2</td><td>1 2</td></tr></table>	$freq(\mathbf{RCL}, B, 3\ 1\ 2) = 2$, $freq(\mathbf{RCL}, B, 1\ 3\ 2) = 1$ $freq(\mathbf{RCL}, C, 1\ 2) = 1$, $freq(\mathbf{RCL}, C, 2\ 1) = 1$ $freq(\mathbf{RCL}, D, 12) = 2$ $prob(\mathbf{RCL}, B, 3\ 1\ 2) = 50\%$, $prob(\mathbf{RCL}, B, 1\ 3\ 2) = 50\%$ $prob(\mathbf{RCL}, C, 1\ 2) = 50\%$, $prob(\mathbf{RCL}, C, 2\ 1) = 50\%$ $prob(\mathbf{RCL}, D, 1\ 2) = 100\%$
$\mathbf{x_2} =$ <table><tr><th></th><th>B</th><th>C</th><th>D</th></tr><tr><td></td><td>1 3 2</td><td>2 1</td><td>1 2</td></tr></table>	

  - Etapa de busca local: adaptação da heurística anterior (BL).

Heurísticas Utilizando Mapeamentos

- Algoritmos Genéticos (AG):
  - População inicial aleatória.
  - Seleção dos k melhores mapeamentos.
  - Novos mapeamentos são criados por mutações e cruzamentos.



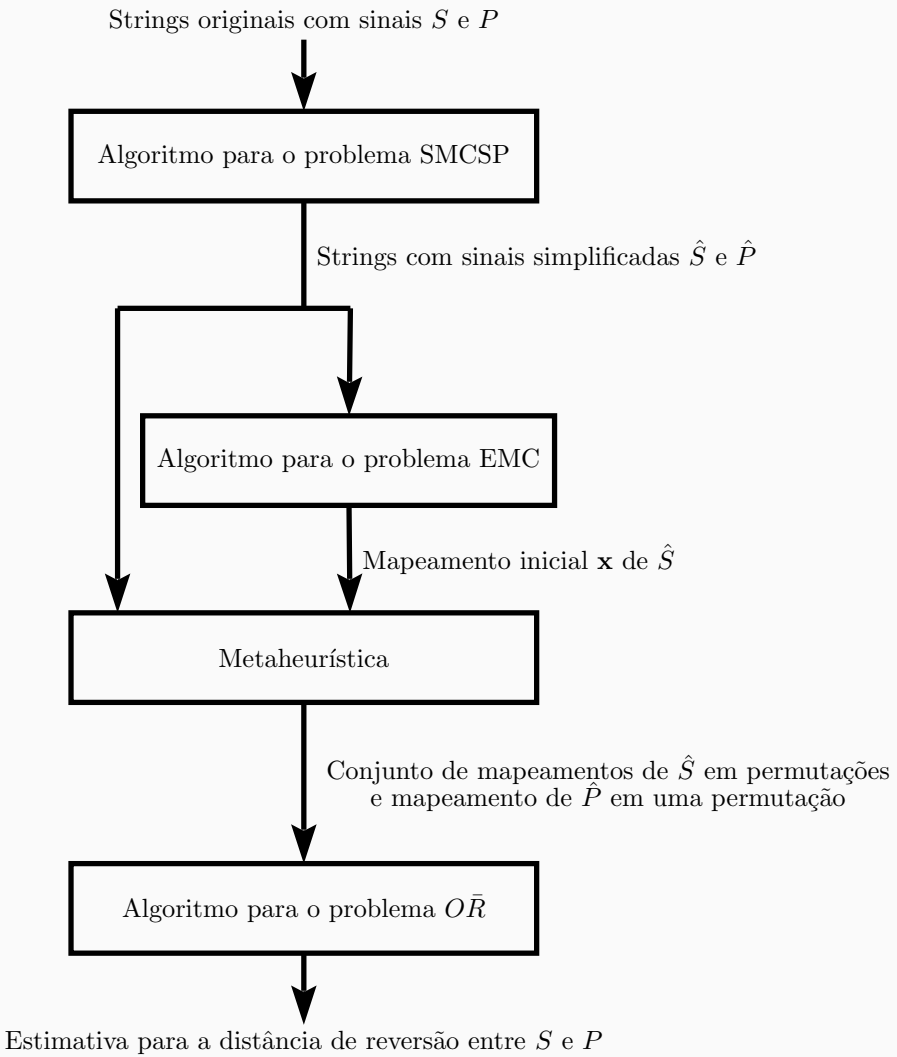


Heurísticas Utilizando Mapeamentos

- Busca Tabu (BT)
- *Simulated Annealing* (SA)
- Busca Cuco (BC)
- Separação (Sep)

Problemas Relacionados

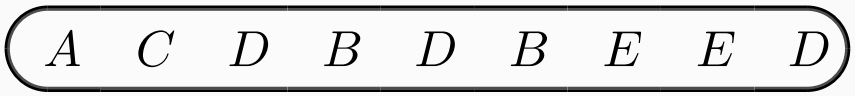
Problemas Relacionados



Partição de Strings Comuns Mínima (MCSP)

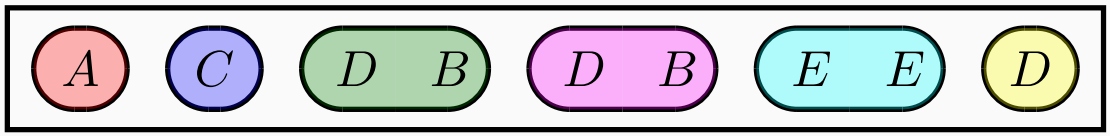
Objetivo: obter a menor **partição direta** $(\mathbb{S}, \mathbb{P}, \phi)$ das strings S e P.

S

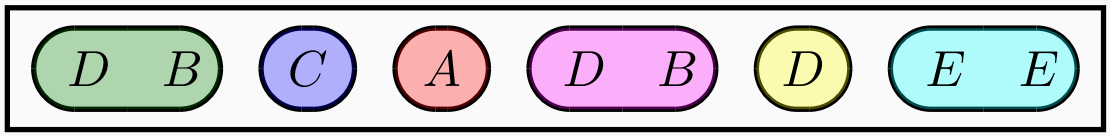


P

$\mathbb{S}$



$\mathbb{P}$

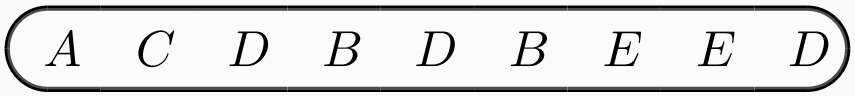


$$\phi = (3\ 2\ 1\ 4\ 6\ 5)$$

Partição de Strings Comuns Mínima (MCSP)

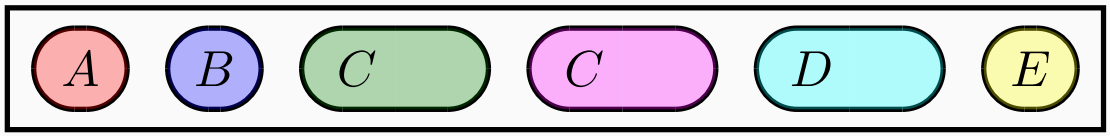
Objetivo: obter a menor **partição direta** $(\mathbb{S}, \mathbb{P}, \phi)$ das strings S e P.

S

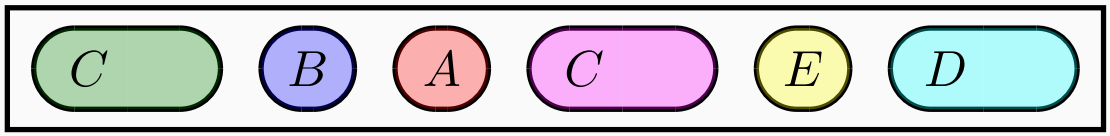


P

$\mathbb{S}$



$\mathbb{P}$



$$\phi = (3\ 2\ 1\ 4\ 6\ 5)$$

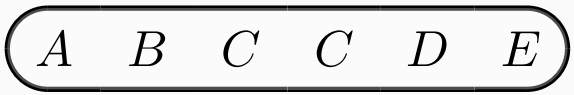
Partição de Strings Comuns Mínima (MCSP)

Objetivo: obter a menor **partição direta** $(\mathbb{S}, \mathbb{P}, \phi)$ das strings S e P.

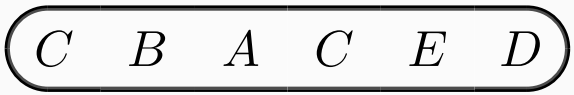
S

P

$\hat{S}$



$\hat{P}$



$$\phi = (3\ 2\ 1\ 4\ 6\ 5)$$

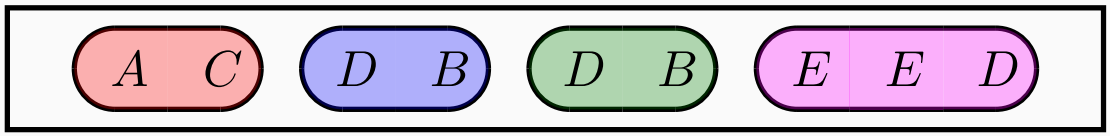
Partição Reversa de Strings Comuns Mínima (RMCSP)

Objetivo: obter a menor **partição reversa** $(\mathbb{S}, \mathbb{P}, \phi)$ das strings S e P.

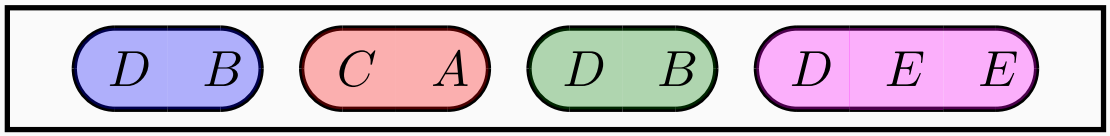
S

P

$\mathbb{S}$



$\mathbb{P}$



$$\phi = (2\ 1\ 3\ 4)$$

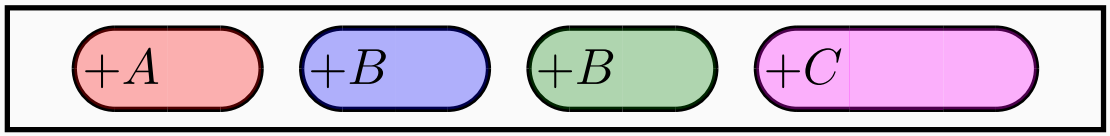
Partição Reversa de Strings Comuns Mínima (RMCSP)

Objetivo: obter a menor **partição reversa** $(\mathbb{S}, \mathbb{P}, \phi)$ das strings S e P.

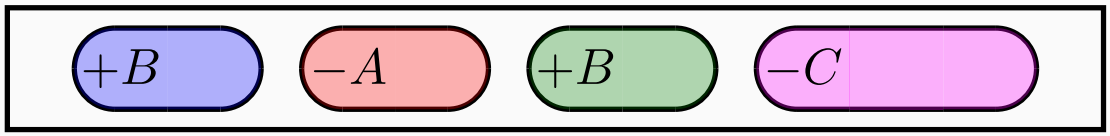
S

P

$\mathbb{S}$



$\mathbb{P}$



$$\phi = (2\ 1\ 3\ 4)$$

Partição Reversa de Strings Comuns Mínima (RMCSP)

Objetivo: obter a menor **partição reversa** $(\mathbb{S}, \mathbb{P}, \phi)$ das strings S e P.

$$S$$

$$A \quad C \quad D \quad B \quad D \quad B \quad E \quad E \quad D$$

$$P$$

$$D \quad B \quad C \quad A \quad D \quad B \quad D \quad E \quad E$$

$$\hat{S}$$

$$+A \quad +B \quad +B \quad +C$$

$$\hat{P}$$

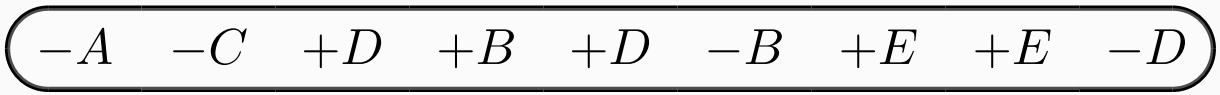
$$+B \quad -A \quad +B \quad -C$$

$$\phi = (2\ 1\ 3\ 4)$$

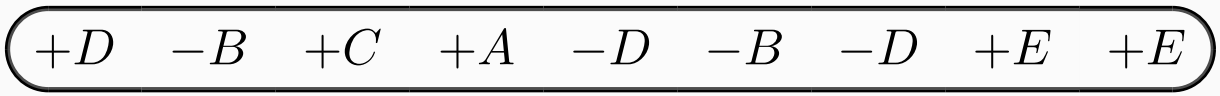
Partição com Sinais de Strings Comuns Mínima (SMCSP)

Objetivo: obter a menor **partição com sinais** $(\mathbb{S}, \mathbb{P}, \phi)$ das strings S e P.

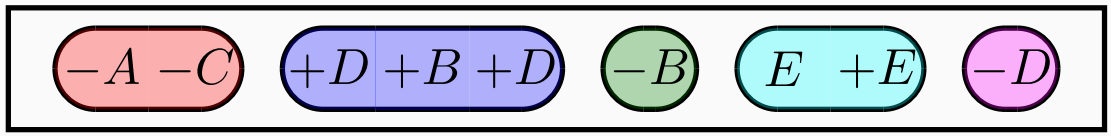
S



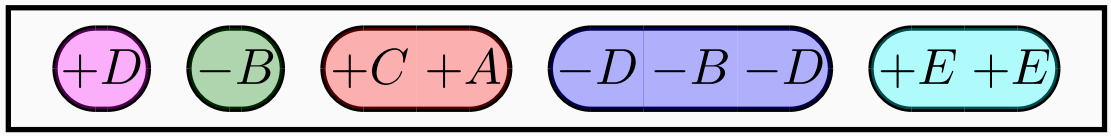
P



$\mathbb{S}$



$\mathbb{P}$

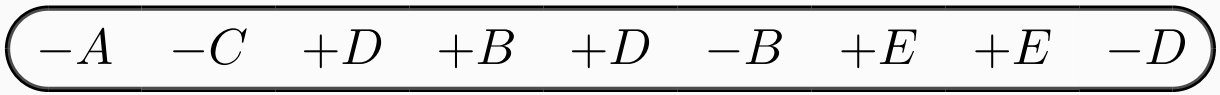


$$\phi = (5\ 3\ 1\ 2\ 4)$$

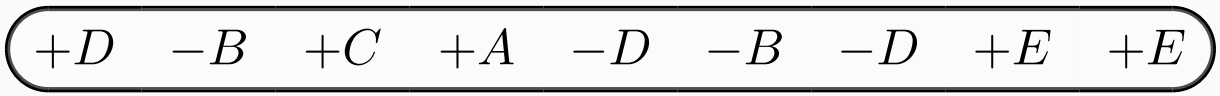
Partição com Sinais de Strings Comuns Mínima (SMCSP)

Objetivo: obter a menor **partição com sinais** $(\mathbb{S}, \mathbb{P}, \phi)$ das strings S e P.

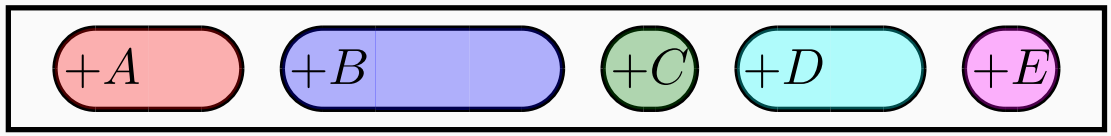
S



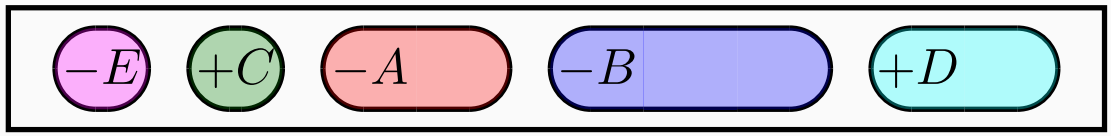
P



$\mathbb{S}$



$\mathbb{P}$



$$\phi = (5\ 3\ 1\ 2\ 4)$$

Partição com Sinais de Strings Comuns Mínima (SMCSP)

Objetivo: obter a menor **partição com sinais** $(\mathbb{S}, \mathbb{P}, \phi)$ das strings S e P.

$$S$$

$$-A \quad -C \quad +D \quad +B \quad +D \quad -B \quad +E \quad +E \quad -D$$

$$P$$

$$+D \quad -B \quad +C \quad +A \quad -D \quad -B \quad -D \quad +E \quad +E$$

$$\hat{S}$$

$$+A \quad +B \quad +C \quad +D \quad +E$$

$$\hat{P}$$

$$-E \quad +C \quad -A \quad -B \quad +D$$

$$\phi = (5\ 3\ 1\ 2\ 4)$$

Relações com Problemas de Rearranjo

Adaptações do Teorema 4.7 de Chen et al. 2005.

Teorema 3

Uma ℓ-aproximação para o problema SMCSP garante uma 2ℓ-aproximação para o problema $D\bar{R}$.

Teorema 4

Uma ℓ-aproximação para o problema RMCSP garante uma 2ℓ-aproximação para o problema DR.

Relações com Problemas de Rearranjo

Teorema 5

Uma ℓ-aproximação para o problema MCSP garante uma 3ℓ-aproximação para o problema DT.

Teorema 6

Uma ℓ-aproximação para o problema SMCSP garante uma 3ℓ-aproximação para o problema $D\bar{R}T$.

Teorema 7

Uma ℓ-aproximação para o problema RMCSP garante uma 3ℓ-aproximação para o problema DRT.

Algoritmos da Literatura

- Os problemas pertencem à classe NP-Difícil (Goldstein et al. 2005).

Algoritmos da Literatura

- Os problemas pertencem à classe NP-Difícil (Goldstein et al. 2005).
- PSOAR (Chen et al. 2005):
  - Feito para o problema SMCSP;
  - Garante uma aproximação com fator 1.5 com genes duplicados;
  - Generalizamos para os outros casos.

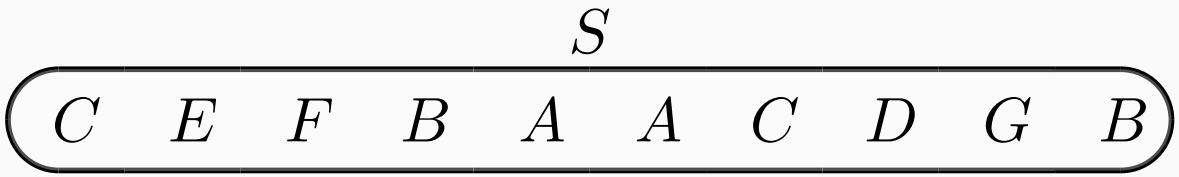
Algoritmos da Literatura

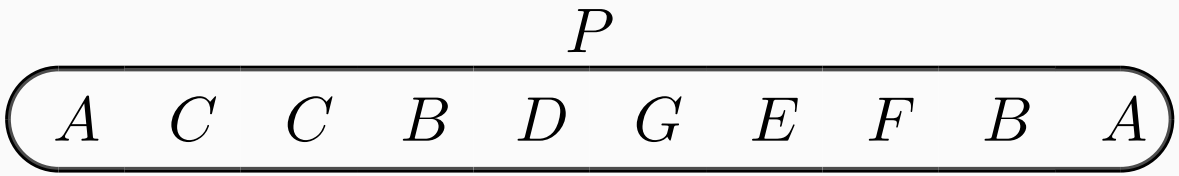
- Os problemas pertencem à classe NP-Difícil (Goldstein et al. 2005).
- PSOAR (Chen et al. 2005):
  - Feito para o problema SMCSP;
  - Garante uma aproximação com fator 1.5 com genes duplicados;
  - Generalizamos para os outros casos.
- HS (Kolman e Waleń 2007a):
  - Feito para o problema MCSP;
  - Garante uma aproximação com fator $4k$;
  - Generalização descrita pelos próprios autores (fator $8k$).

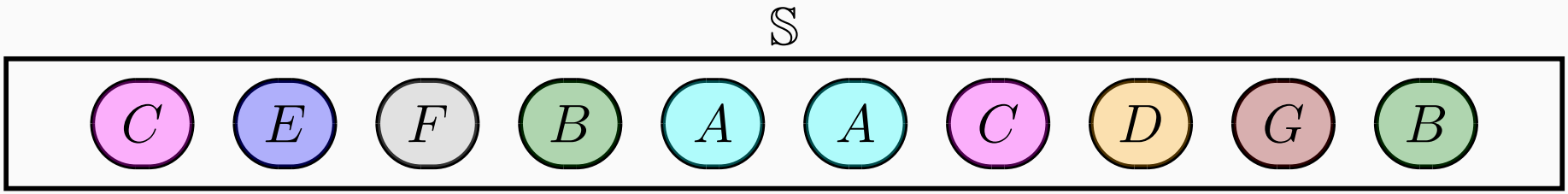
Algoritmos da Literatura

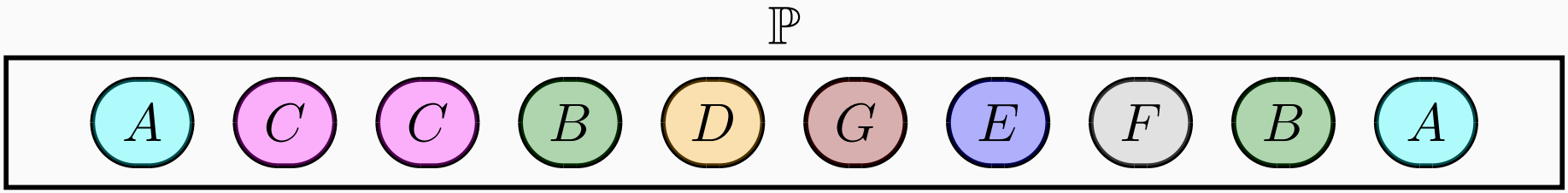
- Os problemas pertencem à classe NP-Difícil (Goldstein et al. 2005).
- PSOAR (Chen et al. 2005):
  - Feito para o problema SMCSP;
  - Garante uma aproximação com fator 1.5 com genes duplicados;
  - Generalizamos para os outros casos.
- HS (Kolman e Waleń 2007a):
  - Feito para o problema MCSP;
  - Garante uma aproximação com fator $4k$;
  - Generalização descrita pelos próprios autores (fator $8k$).
- GREEDY (Chrobak et al. 2004) e GREEDY* (He 2007):
  - Feitos para o problema MCSP;
  - Garantem uma aproximação com fator em $\Omega(n^{0.43})$ e $O(n^{0.69})$;
  - Generalizamos para os outros casos.

Heurística de Combinação

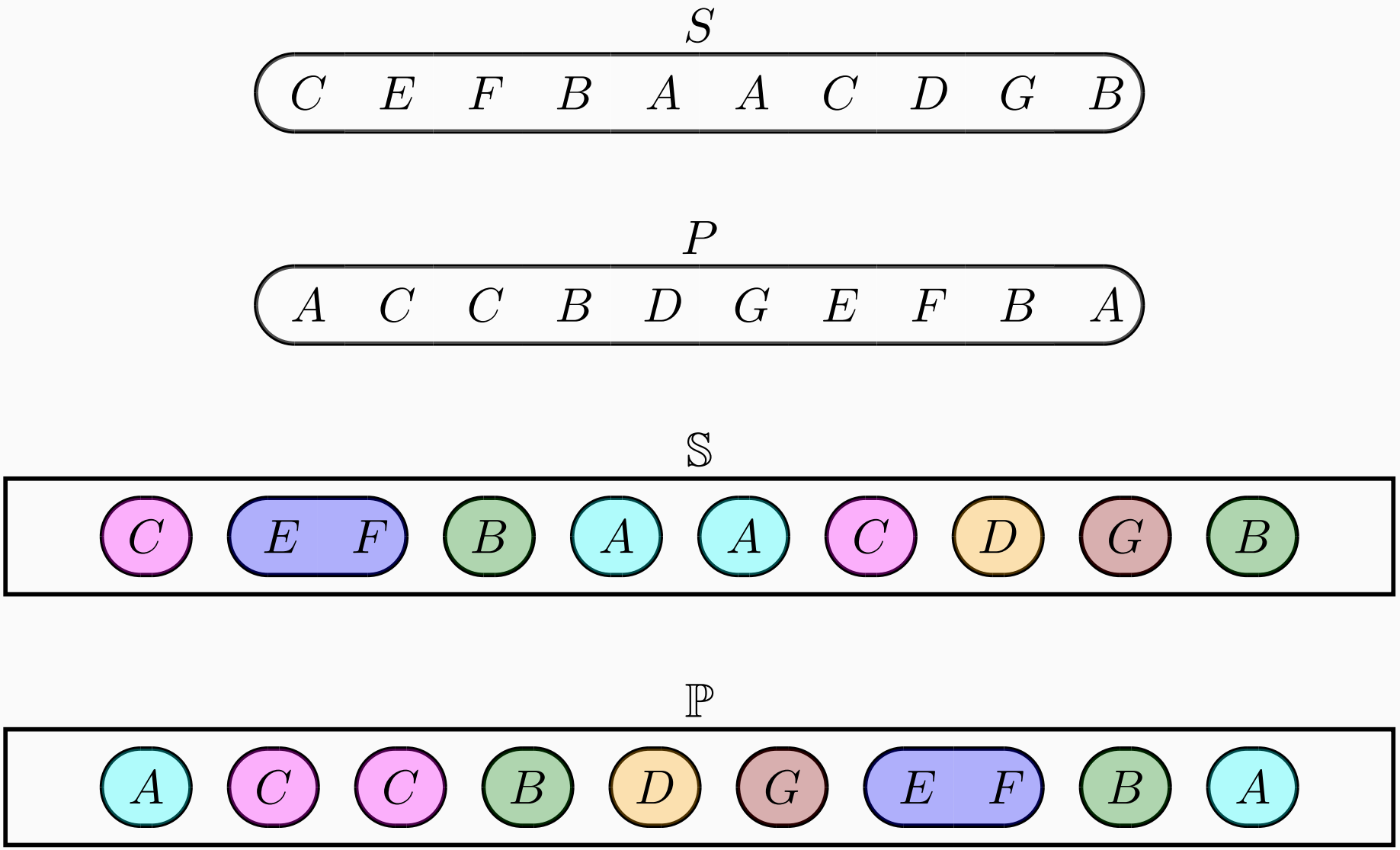




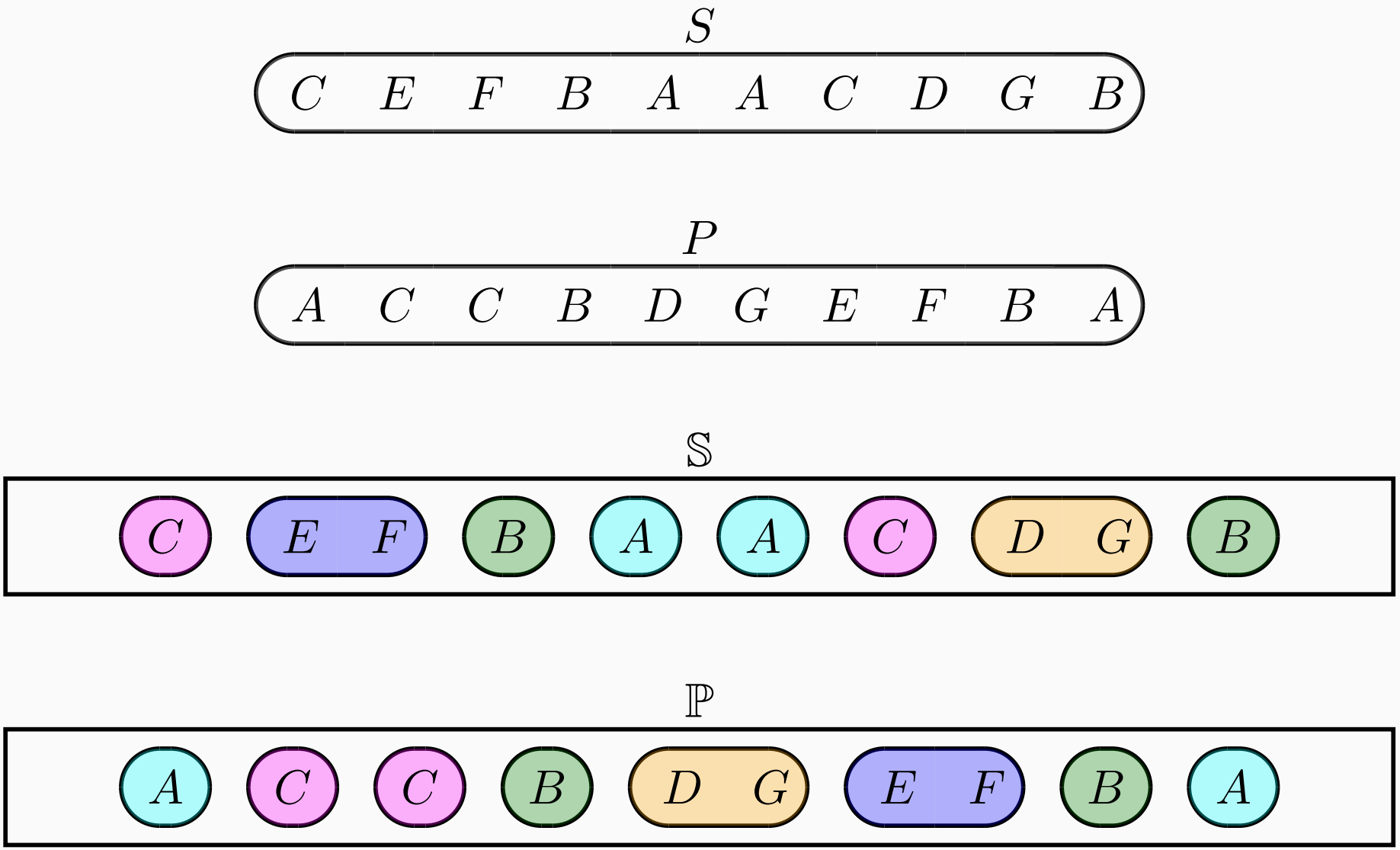




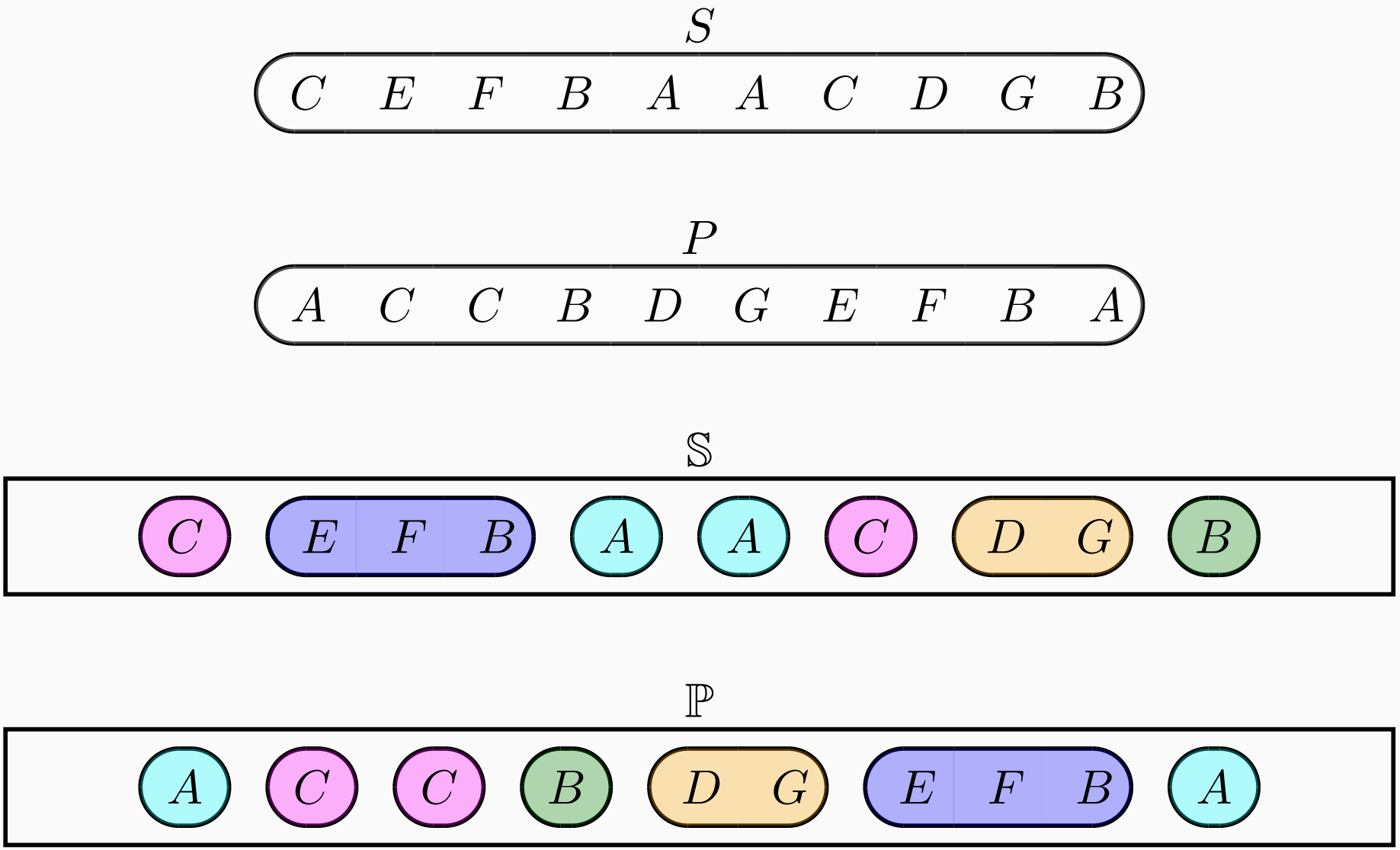
Heurística de Combinação



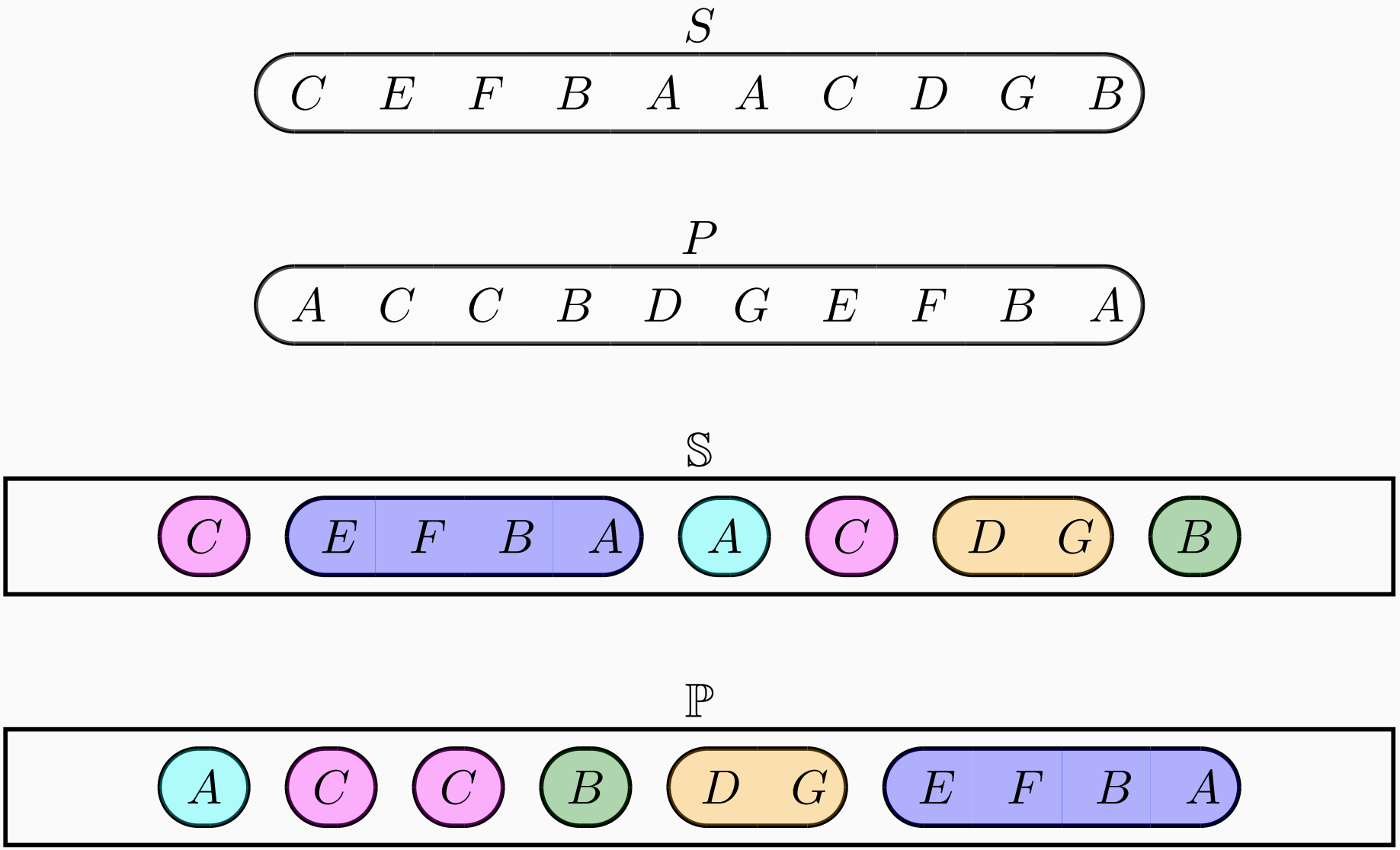
Heurística de Combinação



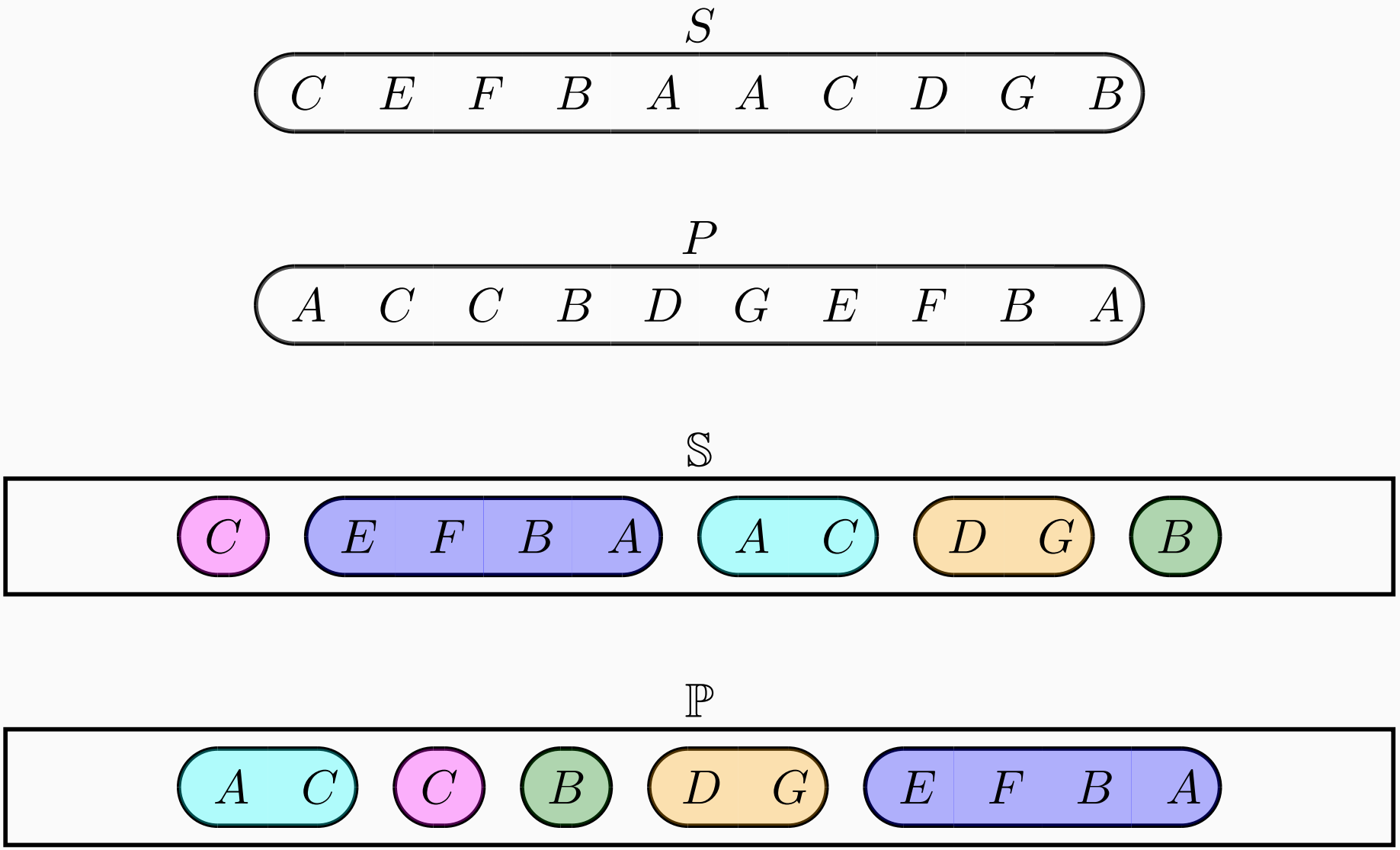
Heurística de Combinação



Heurística de Combinação



Heurística de Combinação



Empacotamento Máximo de Ciclos

$$\hat{S}$$

$$+A \quad +D \quad -B \quad -D \quad -C \quad +C \quad +E$$

$$\hat{P}$$

$$+A \quad +B \quad -D \quad +D \quad -C \quad -C \quad +E$$

$$\tilde{S}$$

$$A \quad D^h \quad D^t \quad B^t \quad B^h \quad D^t \quad D^h \quad C^t \quad C^h \quad C^h \quad C^t \quad E$$

$$\tilde{P}$$

$$A \quad B^h \quad B^t \quad D^t \quad D^h \quad D^h \quad D^t \quad C^t \quad C^h \quad C^t \quad C^h \quad E$$

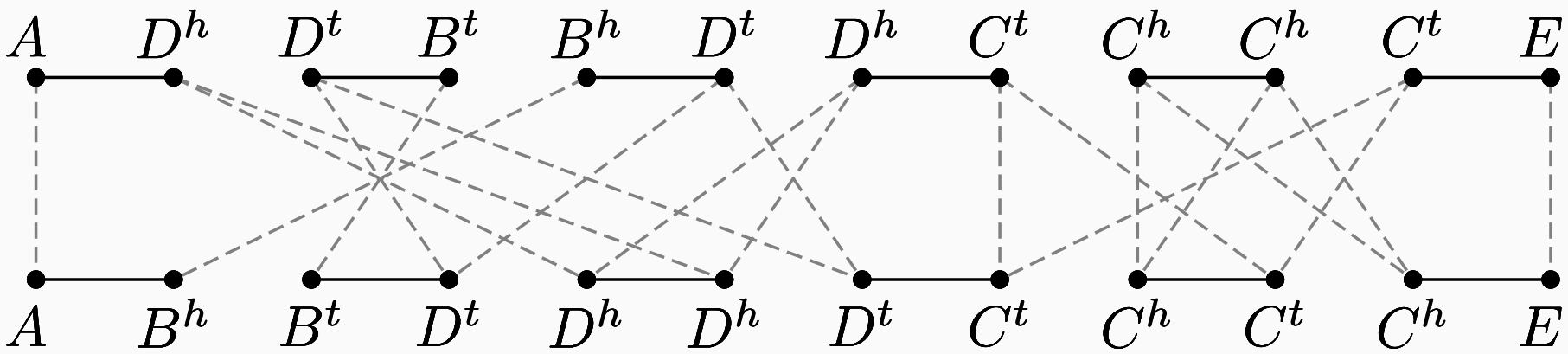
Empacotamento Máximo de Ciclos

$\tilde{S}$

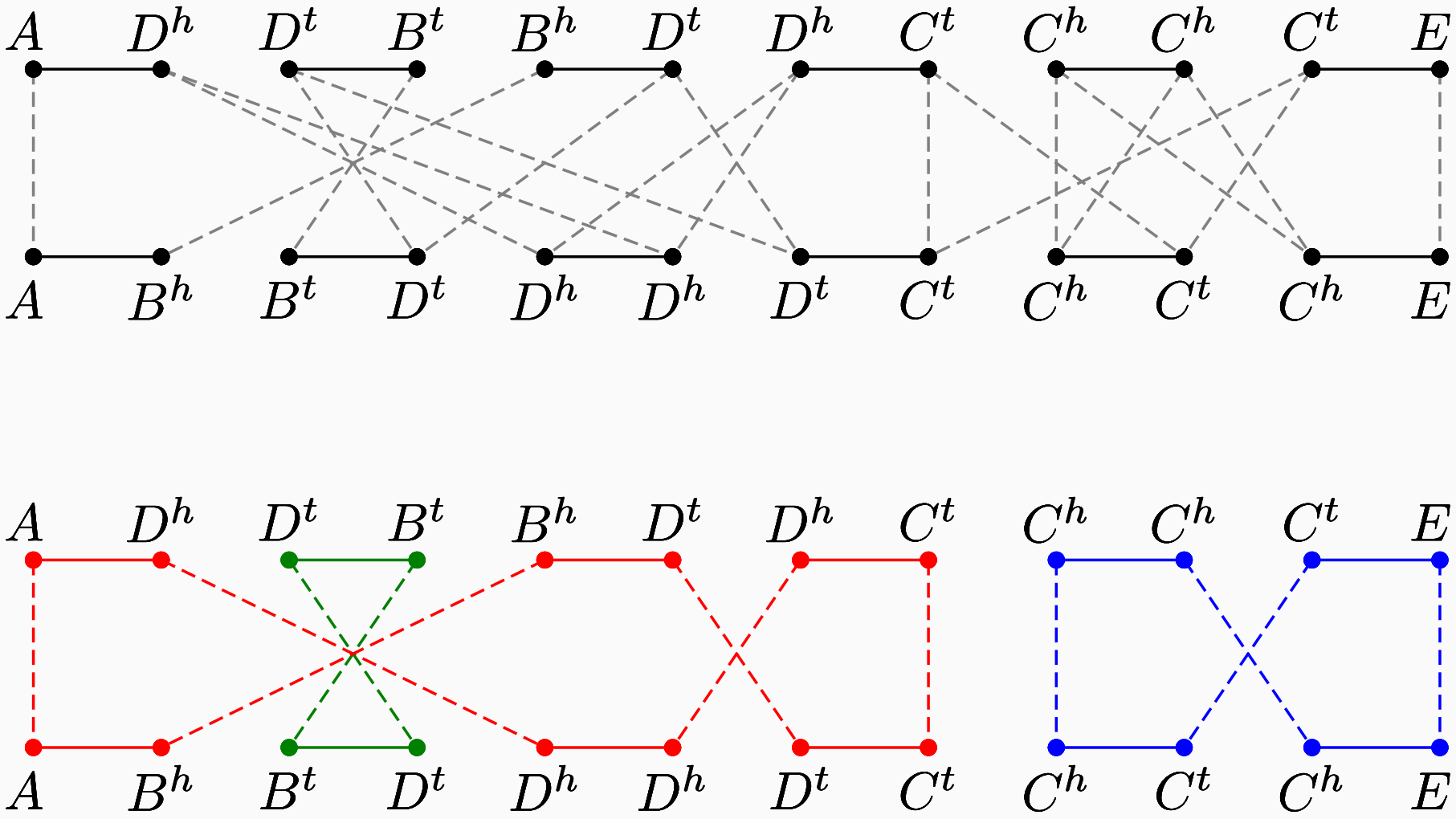
$A \quad D^h \quad D^t \quad B^t \quad B^h \quad D^t \quad D^h \quad C^t \quad C^h \quad C^h \quad C^t \quad E$

$\tilde{P}$

$A \quad B^h \quad B^t \quad D^t \quad D^h \quad D^h \quad D^t \quad C^t \quad C^h \quad C^t \quad C^h \quad E$



Empacotamento Máximo de Ciclos



Experimentos Práticos

Resultados

- Criamos várias bases de dados para testar as heurísticas.
  - Bases com caracteres duplicados para cada problema: REV, SREV, TRANS, REVTRANS e SREVTRANS;
  - Bases difíceis com caracteres duplicados: HARD e SHARD;
  - Bases aleatórias: RAND e SRAND;
  - Bases para cada problema, variando a ocorrência máxima: R-O, SR-O, T-O, RT-O e SRT-O;
  - Bases para cada problema, variando o tamanho do alfabeto: R-L, SR-L, T-L, RT-L e SRT-L;

Resultados

- Executamos testes para selecionar os parâmetros das heurísticas.
- Realizamos uma comparação com alguns algoritmos da literatura para os problemas de distância de rearranjo em permutações.
- Também comparamos os algoritmos para partição e utilizamos os melhores algoritmos encontrados para simplificar as strings das bases contendo caracteres com mais de duas cópias.
- Comparamos com o algoritmo SOAR (Chen et al. 2005) e suas variações.

Resultados

Tabela 1: Sumário dos resultados considerando genes duplicados. Os valores correspondem à média das razões entre as distâncias e o tamanho das strings em cada base de dados.

Problema	Base	MA	BL	GRASP	AG	BT	BC	SA	Sep	SOAR
DR	REV	0.42	0.26	**0.25**	**0.25**	0.26	0.28	**0.25**	0.26	0.32
$D\bar{R}$	SREV	0.41	**0.25**	**0.25**	**0.25**	**0.25**	**0.25**	**0.25**	0.26	**0.25**
DT	TRANS	0.35	0.27	**0.25**	0.27	0.30	0.32	0.31	0.27	0.27
DRT	REVTRANS	0.31	0.23	**0.22**	0.23	0.25	0.27	0.27	0.23	0.25
$D\bar{R}T$	SREVTRANS	0.31	0.23	**0.22**	0.23	0.26	0.27	0.37	0.23	0.24
DR	HARD	0.73	**0.65**	**0.65**	0.66	0.66	0.66	**0.65**	0.69	**0.65**
$D\bar{R}$	SHARD	0.94	0.92	0.92	0.92	0.92	0.93	**0.91**	0.92	0.94
DT	HARD	**0.51**	**0.51**	**0.51**	**0.51**	**0.51**	**0.51**	**0.51**	**0.51**	0.53
DRT	HARD	0.50	**0.49**	**0.49**	**0.49**	**0.49**	**0.49**	**0.49**	**0.49**	0.50
$D\bar{R}T$	SHARD	0.52	**0.51**	**0.51**	**0.51**	**0.51**	**0.51**	**0.51**	**0.51**	0.54

Resultados

Tabela 2: Sumário dos resultados considerando genes multiplicados. Os valores correspondem à média das razões entre as distâncias e o tamanho das strings em cada base de dados.

Problema	Base	MA	BL	GRASP	AG	AG*	BL*	GRASP*	SOAR*
DR	RAND	0.95	0.92	0.93	0.93	0.92	**0.91**	0.92	0.92
$D\bar{R}$	SRAND	0.97	0.95	0.95	0.95	**0.93**	**0.93**	0.94	0.94
DT	RAND	0.54	0.53	0.53	0.53	0.53	**0.52**	0.53	0.55
DRT	RAND	0.49	**0.48**	0.49	**0.48**	**0.48**	**0.48**	**0.48**	0.49
$D\bar{R}T$	SRAND	0.51	0.51	0.51	0.51	0.51	**0.50**	0.51	0.52
DR	R-O	0.59	**0.57**	**0.57**	**0.57**	**0.57**	**0.57**	**0.57**	0.58
$D\bar{R}$	SR-O	0.54	**0.52**	**0.52**	0.53	**0.52**	**0.52**	**0.52**	0.53
DT	T-O	0.40	0.40	0.40	0.40	0.40	**0.39**	0.40	0.42
DRT	RT-O	**0.35**	**0.35**	**0.35**	**0.35**	**0.35**	**0.35**	**0.35**	0.36
$D\bar{R}T$	SRT-O	0.36	**0.35**	0.36	0.36	0.36	**0.35**	0.36	0.37
DR	R-L	0.58	**0.57**	**0.57**	**0.57**	**0.57**	**0.57**	**0.57**	0.58
$D\bar{R}$	SR-L	0.55	0.52	0.52	0.52	0.52	0.52	**0.51**	0.54
DT	T-L	0.40	**0.39**	0.40	0.40	0.40	**0.39**	0.40	0.42
DRT	RT-L	**0.35**	**0.35**	**0.35**	**0.35**	**0.35**	**0.35**	**0.35**	0.36
$D\bar{R}T$	SRT-L	0.36	**0.35**	0.36	0.36	**0.35**	**0.35**	0.36	0.37

Conclusões

Conclusões

- Desenvolvemos heurísticas para explorar de forma eficiente o espaço de mapeamentos possíveis.
- Combinamos as heurísticas com algoritmos da literatura para melhorar a solução no caso geral e manter as mesmas aproximações teóricas já conhecidas.
- Como verificamos pelos experimentos práticos, a heurística Busca Local obteve bons resultados no caso geral.
- Quando estamos lidando apenas com o evento de reversão ou com um limite de duas cópias para cada gene, as heurísticas GRASP e Algoritmo Genético também se mostraram boas alternativas.

Conclusões

- Este trabalho gerou um artigo (Siqueira et al. 2020) publicada nos anais da "7th International Conference on Algorithms for Computational Biology" (AICoB'2020).
- Atualmente, temos uma versão estendida desse trabalho em avaliação para uma edição especial da revista "IEEE/ACM Transactions on Computational Biology and Bioinformatics" (TCBB).
- Pretendemos continuar explorando esse problema utilizando outras abordagens, como novos algoritmos para o problema de Empacotamento Máximo de Ciclos.

Referências

Referências

- Piotr Berman, Sridhar Hannenhalli e Marek Karpinski (2002). "1.375-Approximation Algorithm for Sorting by Reversals". Em: *Proceedings of the 10th Annual European Symposium on Algorithms (ESA'2002)*. London, UK: Springer-Verlag, pp. 200–210.
- Laurent Bulteau, Guillaume Fertin e Irena Rusu (2012). "Sorting by Transpositions Is Difficult". Em: *SIAM Journal on Discrete Mathematics* 26.3, pp. 1148–1180.
- Alberto Caprara (1999). "Sorting Permutations by Reversals and Eulerian Cycle Decompositions". Em: *SIAM Journal on Discrete Mathematics* 12.1, pp. 91–110.
- Xin Chen (2010). "On sorting unsigned permutations by double-cut-and-joins". Em: *Journal of Combinatorial Optimization* 25.3, pp. 339–351.

Referências


Xin Chen, Jie Zheng, Zheng Fu, Peng Nan, Yang Zhong, Stefano Lonardi e Tao Jiang (2005). "Assignment of Orthologous Genes via Genome Rearrangement". Em: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 2.4, pp. 302–315.


Marek Chrobak, Petr Kolman e Jiří Sgall (2004). "The Greedy Algorithm for the Minimum Common String Partition Problem". Em: *Proceedings of the 7th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'2004), and 8th International Workshop on Randomization and Computation (RANDOM'2004)*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 84–95.

Referências


Graham Cormode e S. Muthukrishnan (2007). "The string edit distance matching problem with moves". Em: *ACM Transactions on Algorithms* 3.1, pp. 1–19.


Avraham Goldstein, Petr Kolman e Jie Zheng (2005). "Minimum Common String Partition Problem: Hardness and Approximations". Em: *Proceedings of the 15th International Symposium on Algorithms and Computation (ISAAC'2004)*. Berlin, Heidelberg, pp. 484–495.


Sridhar Hannenhalli e Pavel A. Pevzner (1999). "Transforming cabbage into turnip. polynomial algorithm for sorting signed permutations by reversals". Em: *Journal of ACM* 46.1, pp. 1–27.

Referências

- Dan He (2007). "A Novel Greedy Algorithm for the Minimum Common String Partition Problem". Em: *Bioinformatics Research and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 441–452.
- Petr Kolman e Tomasz Waleń (2007a). "Approximating reversal distance for strings with bounded number of duplicates". Em: *Discrete Applied Mathematics* 155.3, pp. 327–336.
- Petr Kolman e Tomasz Waleń (2007b). "Reversal Distance for Strings with Duplicates: Linear Time Approximation Using Hitting Set". Em: *Proceedings of the 4th International Workshop on Approximation and Online Algorithms (WAOA'2006)*. Vol. 4368. Lecture Notes in Computer Science. Berlin, Heidelberg, pp. 279–289.

Referências


Andre Rodrigues Oliveira, Klairton Lima Brito, Ulisses Dias e Zanoni Dias (2019). "On the Complexity of Sorting by Reversals and Transpositions Problems". Em: *Journal of Computational Biology* 26.11, pp. 1223–1229.

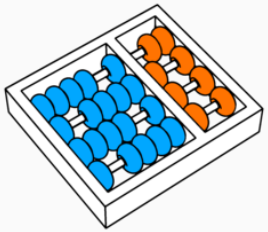

Andrew J. Radcliffe, Alex D. Scott e Elizabeth Wilmer (2005). "Reversals and Transpositions Over Finite Alphabets". Em: *SIAM Journal on Discrete Mathematics* 19.1, pp. 224–244.


Atif Rahman, Swakkhar Shatabda e Masud Hasan (2008). "An approximation algorithm for sorting by reversals and transpositions". Em: *Journal of Discrete Algorithms* 6.3, pp. 449–457.

Referências

Gabriel Siqueira, Klairton Lima Brito, Ulisses Dias e Zanoni Dias (2020). "Heuristics for Reversal Distance Between Genomes with Duplicated Genes". Em: *Proceedings of the 7th International Conference on Algorithms for Computational Biology (AlCoB'2020)*. Cham: Springer International Publishing, pp. 29–40.

Maria Emília M.T. Walter, Zanoni Dias e João Meidanis (1998). "Reversal and transposition distance of linear chromosomes". Em: *Proceedings of the String Processing and Information Retrieval: A South American Symposium (SPIRE'1998)*. Los Alamitos, CA, USA, pp. 96–102.



UNICAMP

Heurísticas para Problemas de Rearranjo de Genomas com Genes Multiplicados

Aluno: Gabriel Henriques Siqueira
Orientador: Prof. Dr. Zanoni Dias
Coorientador: Dr. André Rodrigues Oliveira

Instituto de Computação – Unicamp