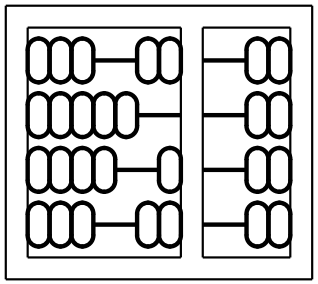
Programação por Restrições aplicada a Problemas de Rearranjo de Genomas

Victor de Abreu Iizuka
Orientador: Zanoni Dias

Instituto de Computação, Universidade Estadual de Campinas



1. Introdução
   - Problema da Distância de Reversão
   - Problema da Distância de Transposição
   - Problema da Distância de Reversão e Transposição
2. Modelos
   - Modelo CSP
   - Modelo COP
3. Análise dos Resultados
   - Especificações Técnicas
   - Descrição dos Testes
   - Resultados
4. Conclusões

Introdução [1/3]

- Rearranjo de genomas tem como o objetivo encontrar o menor número de operações que transformam um genoma em outro.
- Essas operações podem ser reversões, transposições, fissões e fusões.
- Estudos mostram que rearranjos são mais adequados que mutações pontuais quando se deseja comparar os genomas de duas espécies [Palmer e Herbon (1988), Bafna e Pevzner (1995)].

Introdução [2/3]

- Neste contexto, a distância evolutiva é o menor número de operações que são necessárias para transformar um genoma em outro.
- Neste trabalho, trataremos os casos em que os eventos de reversões e transposições ocorrem de forma isolada e os casos quando os dois eventos ocorrem ao mesmo tempo.
- O trabalho desenvolvido nesta dissertação segue a linha de pesquisa utilizada por Dias e Dias (2009) e nós apresentaremos modelos de Programação por Restrições (PR/CP) para ordenação por reversões e ordenação por reversões e transposições, baseados na teoria do Problema de Satisfação de Restrições (PSR/CSP) e na teoria do Problema de Otimização com Restrições (POR/COP).

Introdução [3/3]

- Nós fizemos comparações com os modelos de Programação por Restrições para ordenação por transposições, descrito por Dias e Dias (2009), e no caso de ordenação por reversões e ordenação por reversões e transposições, com os modelos de Programação Linear Inteira (PLI/ILP) descritos por Dias e Souza (2007).
- O artigo *Constraint logic programming models for reversal and transposition distance problems* foi publicado no *VI Brazilian Symposium on Bioinformatics* (BSB'2011).

Permutação

- Para fins computacionais, um genoma é representado por uma n-tupla de genes, e quando não há genes repetidos essa tupla é chamada de permutação.
- Uma permutação é representada como $\pi = (\pi_1\ \pi_2\ \ldots\ \pi_n)$, para $\pi_i \in \mathbb{N}$, $1 \leq \pi_i \leq n$ e $i \neq j \leftrightarrow \pi_i \neq \pi_j$.
- A permutação identidade é representada como $\iota = (1\ 2\ 3\ \ldots\ n)$.

Problema da Distância de Reversão [1/2]

Evento de reversão:

Um evento de reversão ocorre quando um bloco do genoma é invertido.

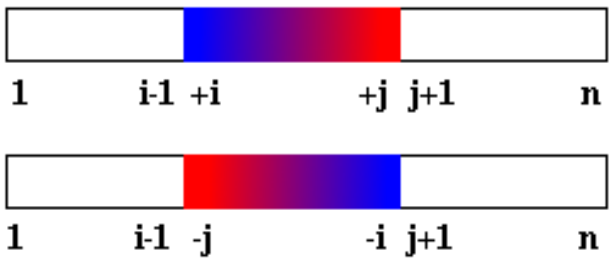


Figura : Uma reversão aplicada sobre uma permutação orientada.

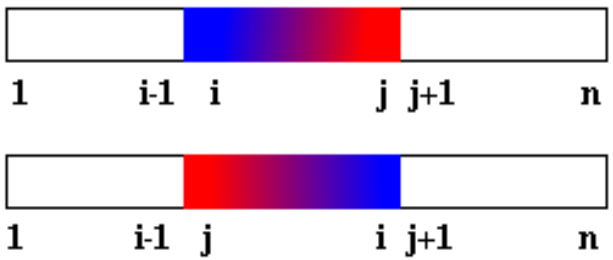


Figura : Uma reversão aplicada sobre uma permutação não orientada.

Problema da Distância de Reversão [2/2]

- O Problema da Distância de Reversão é encontrar o número mínimo de reversões necessárias para transformar uma genoma em outro.
- O Problema da Distância de Reversão é equivalente ao problema de Ordenação por Reversões, que é a distância de reversão entre a permutação π e a permutação identidade ι, denotado por $d_r(\pi)$.
- Se a orientação dos genes é conhecida, o problema pode ser solucionado com algoritmos polinomiais.
- Caso contrário, o problema pertence à classe de problemas NP-Difíceis, com a prova apresentada por Caprara (1997).
- Melhor algoritmo de aproximação possui razão de 1.375 e foi apresentado por Berman, Hannenhalli e Karpinski (2002).

Grafo de Breakpoints para Reversões [1/4]

Breakpoints:

Dois elementos consecutivos π_i e π_{i+1}, $0 \leq i \leq n$, são *adjacentes* quando $|\pi_i - \pi_{i+1}| = 1$, e são *breakpoints* caso contrário.

- A permutação π é estendida adicionando os elementos $\pi_0 = 0$ e $\pi_{n+1} = n + 1$.
- Grafo de arestas coloridas $G(\pi)$ com $n+2$ vértices $\{0, 1, \ldots, n, n+1\}$.
- Uma aresta preta liga os vértices i e j se (i, j) for um *breakpoint*.
- Uma aresta cinza liga os vértices i e j se $|i - j| = 1$ e os dois não são consecutivos em π.

Grafo de Breakpoints para Reversões [2/4]

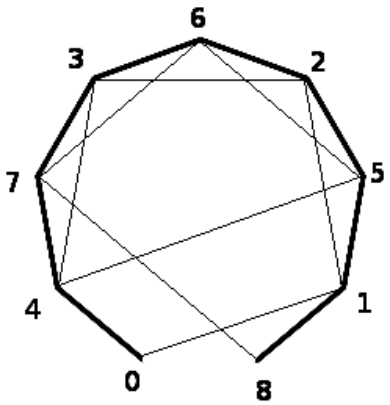


Figura : Grafo de *Breakpoints* da permutação $\pi = (4\ 7\ 3\ 6\ 2\ 5\ 1)$

- Reversão atua em dois pontos em uma permutação, portanto pode reduzir o número de breakpoints em pelo menos um e no máximo dois.

Teorema 1:

Para qualquer permutação π,

$$\frac{1}{2} b_r(\pi) \leq d_r(\pi) \leq b_r(\pi).$$

Grafo de Breakpoints para Reversões [3/4]

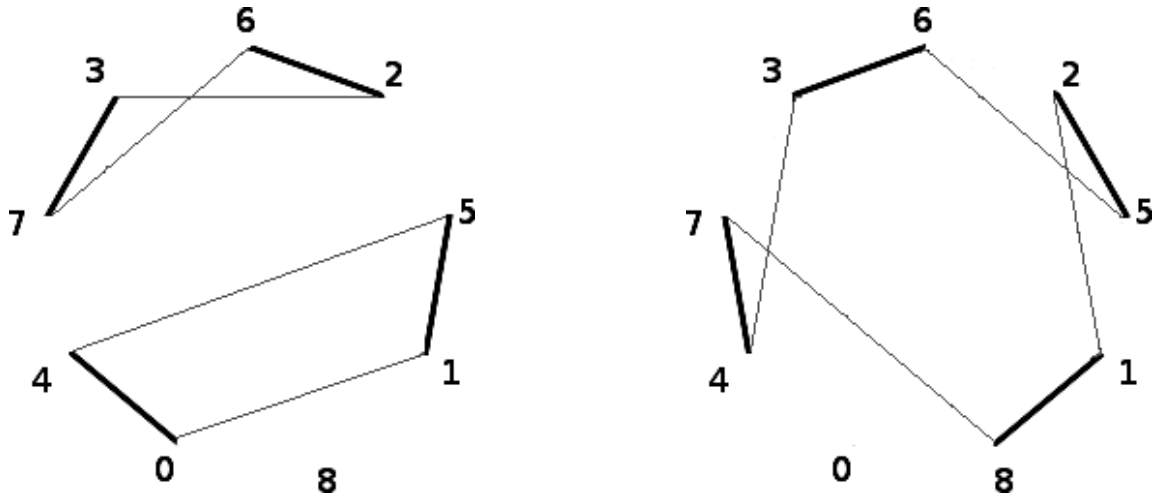


Figura : Exemplo de decomposição em ciclos de arestas disjuntas para o grafo de *breakpoints* da permutação $\pi = (4\ 7\ 3\ 6\ 2\ 5\ 1)$.

Grafo de Breakpoints para Reversões [4/4]

- O grafo pode ser decomposto em ciclos de arestas disjuntas.
- Existem diversas maneiras de realizar a decomposição.
- O Teorema 2, demonstrado no trabalho de Christie (1998), fornece os limitantes para a distância de reversão usando a quantidade de 2-ciclos na máxima decomposição de ciclos de $G(\pi)$.

Teorema 2:

Se $c_2(\pi)$ é o número mínimo de 2-ciclos em qualquer máxima decomposição em ciclos de $G(\pi)$ então:

$$\frac{2}{3}b_r(\pi) - \frac{1}{3}c_2(\pi) \leq d_r(\pi) \leq b_r(\pi) - \frac{1}{2}c_2(\pi).$$

Problema da Distância de Transposição [1/2]

Evento de transposição:

Um evento de transposição ocorre quando dois blocos adjacentes no genoma trocam de posição.

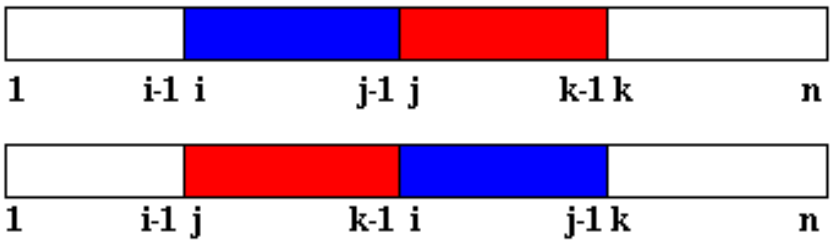


Figura : Uma transposição aplicada em uma permutação.

Problema da Distância de Transposição [2/2]

- O Problema da Distância de Transposição é encontrar o número mínimo de transposições necessárias para transformar um genoma em outro.
- O Problema da Distância de Transposição é equivalente ao problema de Ordenação por Transposições, que é a distância de transposição entre a permutação π e a permutação identidade ι, denotado por $d_t(\pi)$.
- Pertence à classe de problemas NP-Difíceis, a prova foi apresentada por Bulteau, Fertin e Rusu (2010).
- O melhor algoritmo de aproximação possui razão de 1.375 e foi apresentado por Elias e Hartman (2006).

Breakpoints para Transposições

Breakpoints:

Um *breakpoint* é um par (π_i, π_{i+1}), tal que $\pi_{i+1} \neq \pi_i + 1$.

- A permutação π é estendida adicionando os elementos $\pi_0 = 0$ e $\pi_{n+1} = n + 1$.
- Uma transposição atua em três pontos de uma permutação, logo pode reduzir o número de *breakpoints* em pelo menos e no máximo três.

Teorema 3:

Para qualquer permutação π,

$$\frac{1}{3} b_t(\pi) \leq d_t(\pi) \leq b_t(\pi).$$

Grafo de Ciclos [1/3]

- Introduzido por Bafna e Pevzner (1998).
- Grafo direcionado com arestas coloridas $G(\pi)$ com $n+2$ vértices $\{0,\ 1,\ \ldots,\ n,\ n+1\}$.
- Arestas cinzas são direcionadas de $i-1$ para i.
- Arestas pretas são direcionadas de π_i para π_{i-1}.

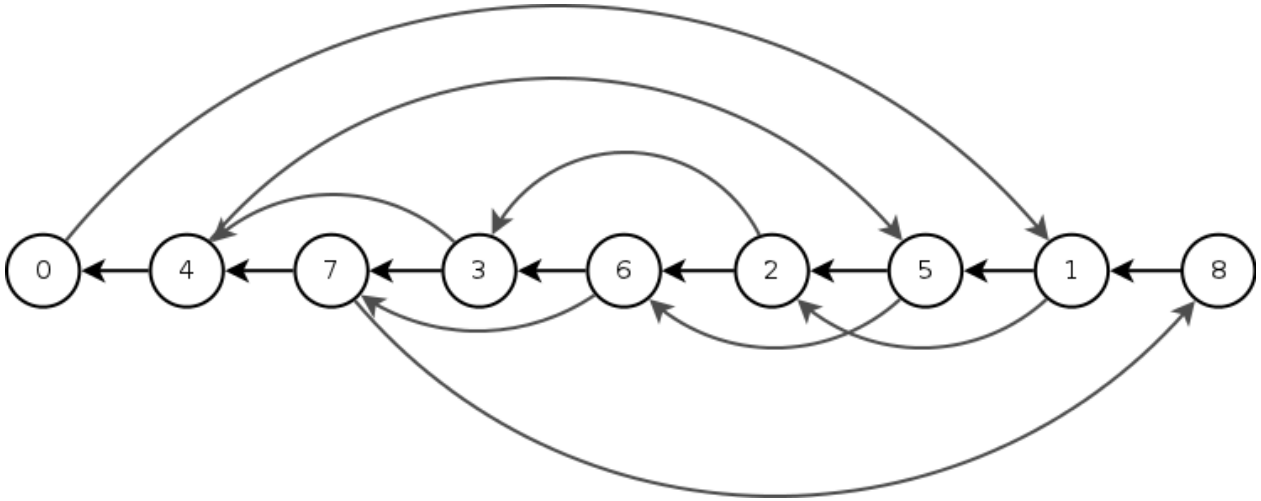


Figura : Grafo de ciclos da permutação $\pi = (4\ 7\ 3\ 6\ 2\ 5\ 1)$

Grafo de Ciclos [2/3]

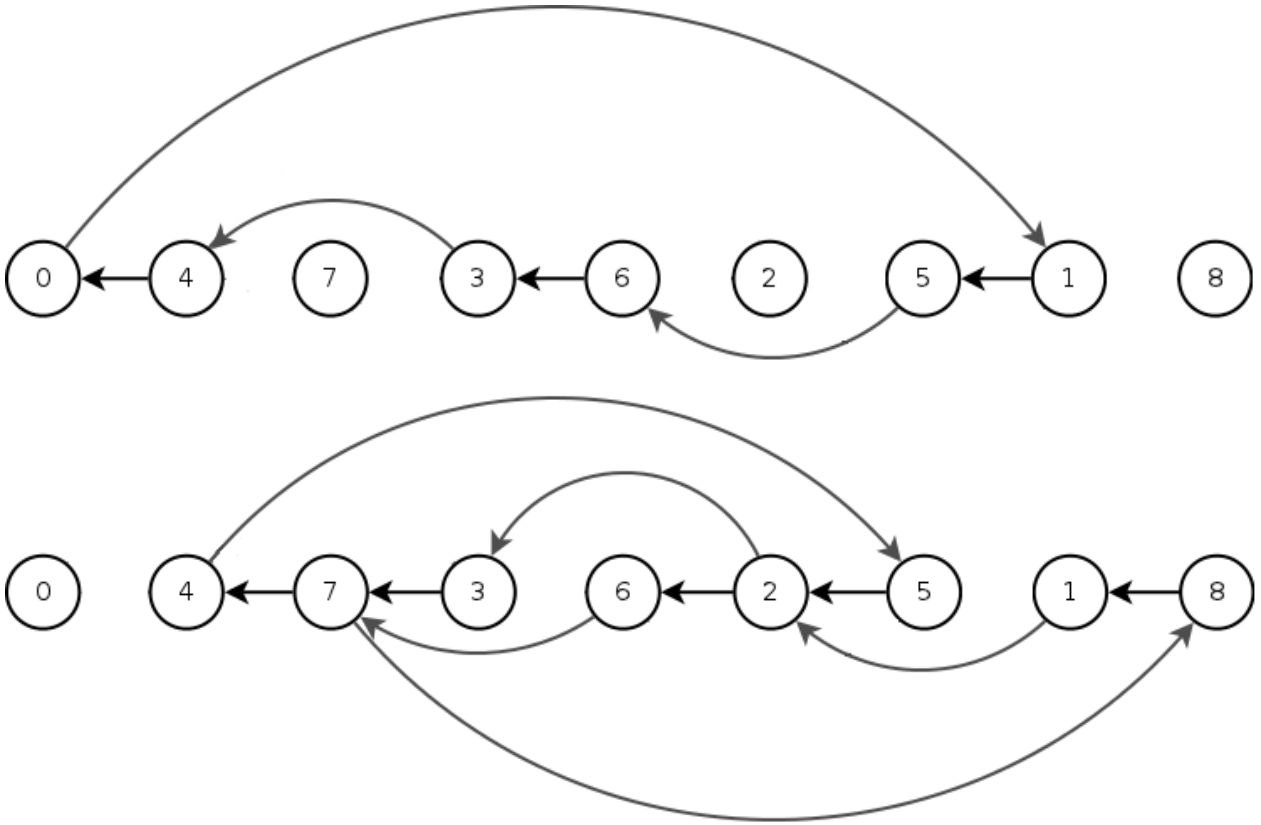


Figura : Exemplo de decomposição em ciclos de arestas disjuntas para o grafo de ciclos da permutação $\pi = (4\ 7\ 3\ 6\ 2\ 5\ 1)$.

Grafo de Ciclos [3/3]

- Para todo vértice de $G(\pi)$ toda aresta chegando é unicamente pareada com uma aresta saindo de cor diferente.
- Existe uma decomposição única de ciclos alternados do conjunto de arestas de $G(\pi)$.
- Seja $c_{\text{ímpar}}(\pi)$ o número de ciclos ímpares de $G(\pi)$, para uma permutação π, e $\Delta c_{\text{ímpar}}(\rho) = c_{\text{ímpar}}(\pi\rho) - c_{\text{ímpar}}(\pi)$ a mudança no número de ciclos ímpares devido a transposição ρ, temos que $\Delta c_{\text{ímpar}} \in \{2, 0, -2\}$ [Bafna e Pevzner (1998)].

Teorema 4:

Para qualquer permutação π,

$$\frac{1}{2}(n + 1 - c_{\text{ímpar}}(\pi)) \leq d_t(\pi) \leq \frac{3}{4}(n + 1 - c_{\text{ímpar}}(\pi)).$$

Problema da Distância de Reversão e Transposição

- Na natureza um genoma não sofre apenas eventos de reversão ou transposição de forma isolada.
- O Problema da Distância de Reversão e Transposição é encontrar o número mínimo de reversões e transposições necessárias para transformar um genoma em outro.
- Hannenhalli, Chappey, Koonin e Pevzner (1995), Walter, Dias e Meidanis (1998, 2002), Gu, Peng e Sudborough (1999) e Lin e Xue (1999) estudaram este problema.

Predicados Básicos [1/3]

Permutação:

A permutação π é uma lista de elementos $(\pi_1, \pi_2, \cdots, \pi_n)$ onde $\pi_i \in \mathbb{N}$, $1 \leq \pi_i \leq n$ e $\pi_i \neq \pi_j$ para $i \neq j$. A permutação identidade ι é definida como $\iota = (1\ 2\ 3\ \cdots\ n)$.

permutation/2

$$
\begin{aligned}
&\mathit{permutation}(\pi, N)\ \text{:-}\\
&\quad \mathit{length}(\pi, N),\\
&\quad \pi\ ::\ [1\ ..\ N],\\
&\quad \mathit{all\_different}(\pi).
\end{aligned}
$$

Predicados Básicos [2/3]

Reversão:

Uma reversão $\rho(i, j)$, $0 < i < j \leq n$, divide a lista em três sublistas $C_1 C_2 C_3$ onde $C_1 = (\pi_1 \; .. \; \pi_{i-1})$, $C_2 = (\pi_i \; .. \; \pi_j)$ e $C_3 = (\pi_{j+1} \; .. \; \pi_n)$. Aplicamos a reversão na sublista C_2, resultando na sublista R_{C_2}. Então juntamos a sublista R_{C_2} com as sublistas C_1 e C_3 para formar $\pi\rho = C_1 R_{C_2} C_3$.

reversal/4

$$
\begin{aligned}
&\mathit{reversal}(\pi, \sigma, I, J) \text{ :-} \\
&\quad \mathit{permutation}(\pi, N), \\
&\quad \mathit{permutation}(\sigma, N), \\
&\quad 1 \leq I < J \leq N, \\
&\quad \mathit{split}(\pi, I, J, C_1, C_2, C_3), \\
&\quad \mathit{reverse}(C_2, R_{C_2}), \\
&\quad \sigma = C_1, R_{C_2}, C_3.
\end{aligned}
$$

Predicados Básicos [3/3]

Transposição:

Uma transposição $\rho(i, j, k)$, $0 < i < j < k \leq n$, divide a lista em quatro sublistas $C_1 C_2 C_3 C_4$ onde $C_1 = (\pi_1 \; .. \; \pi_{i-1})$, $C_2 = (\pi_i \; .. \; \pi_{j-1})$, $C_3 = (\pi_j \; .. \; \pi_{k-1})$ e $C_4 = (\pi_k \; .. \; \pi_n)$. Então juntamos as sublistas para formar $\pi\rho = C_1 C_3 C_2 C_4$. Observe que C_1 e C_4 podem ser vazias.

transposition/5

$$
\begin{aligned}
&\mathit{transposition}(\pi, \sigma, I, J, K) \text{ :-} \\
&\quad \mathit{permutation}(\pi, N), \\
&\quad \mathit{permutation}(\sigma, N), \\
&\quad 1 \leq I < J < K \leq N, \\
&\quad \mathit{split}(\pi, I, J, K, C_1, C_2, C_3, C_4), \\
&\quad \sigma = C_1, C_3, C_2, C_4.
\end{aligned}
$$

Modelo CSP [1/5]

- Primeiramente modelamos os problemas usando a teoria CSP.
- O número de variáveis é desconhecido, porque precisamos do valor da distância $d(\pi)$ para criar as restrições e variáveis que representam as permutações.
- Escolhemos um valor candidato para a distância D tal que $D \in [LB \,..\, UB]$ e tentamos achar uma combinação apropriada de D operações que solucionam o problema.
- Se o CSP falha, escolhemos outro valor para D apenas incrementando o seu valor.
- O valor de D é escolhido usando a estratégia *bottom-up*.

Modelo CSP [2/5]

reversal_distance/3

$reversal\_distance(\iota, 0, \_Model).$

$reversal\_distance(\pi, R, Model)$:-

$bound(\pi, Model, LB, UB),$

$R :: [LB \,..\, UB],$

$indomain(R),$

$reversal(\pi, \sigma, \_I, \_J),$

$reversal\_distance(\sigma, R - 1, Model).$

transposition_distance/3

$transposition\_distance(\iota, 0, \_Model).$

$transposition\_distance(\pi, T, Model)$:-

$bound(\pi, Model, LB, UB),$

$T :: [LB \,..\, UB],$

$indomain(T),$

$transposition(\pi, \sigma, \_I, \_J, \_K),$

$transposition\_distance(\sigma, T - 1, Model).$

Modelo CSP [3/5]

rev_trans_dist/3

$$
\begin{aligned}
&rev\_trans\_dist(\iota, 0, \_Model).\\
&rev\_trans\_dist(\pi, N, Model) :-\\
&\quad bound(\pi, Model, LB, UB),\\
&\quad N :: [LB \,..\, UB],\\
&\quad indomain(N),\\
&\quad event(\pi, \sigma),\\
&\quad rev\_trans\_dist(\sigma, N-1, Model).
\end{aligned}
$$

Modelo CSP [4/5]

Limitantes [1/2]:

- **def_csp** não usa nenhum limitante.
- **rev_br_csp**: Modelo que usa o conceito de *breakpoints* em reversões para calcular os limitantes descritos no Teorema 1.
- **rev_cg_csp**: Modelo que usa o número de 2-ciclos na máxima decomposição em ciclos de $G(\pi)$ para calcular os limitantes descritos no Teorema 2.
- **tra_br_csp**: Modelo que usa o conceito de *breakpoints* em transposições para calcular os limitantes conforme descrito no Teorema 3.
- **tra_cg_csp**: Modelo que usa o conceito de grafo de ciclos em transposições, fazendo a decomposição de ciclos e analisando os ciclos ímpares separadamente para calcular os limitantes conforme descrito no Teorema 4.

Modelo CSP [5/5]

Limitantes [2/2]:

- **r_t_br_csp**: Melhor limitante superior entre o limitante de *breakpoints* para reversões e o limitante de *breakpoints* para transposições.
- **r_t_bc_csp**: Melhor limitante superior entre o limitante de *breakpoints* para reversões e o limitante do grafo de ciclos para transposições.
- **r_t_cc_csp**: Melhor limitante superior entre o limitante do grafo de ciclos para reversões e o limitante do grafo de ciclos para transposições.

Modelo COP [1/7]

- Outra alternativa é modelar o problema usando a teoria COP.
- Esta abordagem necessita de um limitante superior.
- Variáveis binárias B indica quando o evento modificou ou não a permutação fornecida.
- Os limitantes são os mesmos usados para os modelos CSP, modificados para os modelos COP. Então temos os seguintes limitantes: **def_cop**, **rev_br_cop**, **rev_cg_cop**, **tra_br_cop**, **tra_cg_cop**, **r_t_br_cop**, **r_t_bc_cop** e **r_t_cc_cop**.

Modelo COP [2/7]

Reversão:

Dado uma reversão $\rho(i, j)$, adicionamos uma nova restrição para permitir $(i, j) = (0, 0)$. Se $(i, j) = (0, 0)$ então $\pi\rho = \pi$.

Transposição:

Dado uma transposição $\rho(i, j, k)$, adicionamos uma nova restrição para permitir $(i, j, k) = (0, 0, 0)$. Se $(i, j, k) = (0, 0, 0)$ então $\pi\rho = \pi$.

reversal_cop/5

$$reversal\_cop(\iota, \iota, 0, 0, 0).$$
$$reversal\_cop(\pi, \sigma, I, J, 1) \text{ :-}$$
$$reversal(\pi, \sigma, I, J).$$

transposition_cop/6

$$transposition\_cop(\iota, \iota, 0, 0, 0, 0).$$
$$transposition\_cop(\pi, \sigma, I, J, K, 1) \text{ :-}$$
$$transposition(\pi, \sigma, I, J, K).$$

Modelo COP [3/7]

- Predicados que calculam as distâncias foram modificados para utilizar as variáveis binárias B.
- Ajusta os valores de B usando o limitante superior e restringe as permutações fazendo $\pi_k = \pi_{k-1}\rho_k$.
- *length/2* predicado interno do prolog usado para criar uma lista de variáveis não instanciadas com o tamanho dado.
- A função de custo *Cost* é a soma das variáveis B associadas com cada ρ_k, $Cost = \sum_{k=1}^{UB} B_k$.
- O valor da distância é o valor mínimo da função de custo $d = \min Cost$.

Modelo COP [4/7]

upperbound_constraint_rev/4:

O predicado *upperbound_constraint_rev/4* aplica na permutação os efeitos de ρ_k e retorna o valor apropriado de B para cada reversão ρ_k.

reversal_distance_cop/3

$$
\begin{aligned}
&\mathit{reversal\_distance\_cop}(\pi, R, \mathit{Model}) :- \\
&\quad \mathit{bound}(\pi, \mathit{Model}, LB, UB), \\
&\quad \mathit{length}(B, UB), \\
&\quad \mathit{upperbound\_constraint\_rev}(\pi, B, \mathit{Model}, UB), \\
&\quad \mathit{sum}(B, \mathit{Cost}), \\
&\quad \mathit{Cost} \geq LB, \\
&\quad \mathit{minimize}(\mathit{Cost}, R).
\end{aligned}
$$

Modelo COP [5/7]

upperbound_constraint_trans/4:

O predicado *upperbound_constraint_trans/4* aplica na permutação os efeitos de ρ_k e retorna o valor apropriado de B para cada transposição ρ_k.

transposition_distance_cop/3

$transposition\_distance\_cop(\pi, T, Model)$:-
$bound(\pi, Model, LB, UB)$,
$length(B, UB)$,
$upperbound\_constraint\_trans(\pi, B, Model, UB)$,
$sum(B, Cost)$,
$Cost \geq LB$,
$minimize(Cost, T)$.

Modelo COP [6/7]

upperbound_constraint_event/4:

O predicado *upperbound_constraint_event/4* escolhe o melhor evento entre a reversão, usando o predicado *upperbound_constraint_rev/4*, e a transposição, usando o predicado *upperbound_constraint_trans/4*, para minimizar o valor da distância.

rev_trans_dist_cop/3

$$
\begin{aligned}
&rev\_trans\_dist\_cop(\pi, N, Model) :- \\
&\quad bound(\pi, Model, LB, UB), \\
&\quad length(B, UB), \\
&\quad upperbound\_constraint\_event(\pi, B, Model, UB), \\
&\quad sum(B, Cost), \\
&\quad Cost \geq LB, \\
&\quad minimize(Cost, N).
\end{aligned}
$$

Modelo COP [7/7]

upperbound_constraint_rev/4

$$
\begin{aligned}
&upperbound\_constraint\_rev(\iota, [\,], \_Model, \_UB).\\
&upperbound\_constraint\_rev(\pi, [B|Bt], Model, UB) :-\\
&\quad reversal\_cop(\pi, \sigma, \_I, \_J, B),\\
&\quad bound(\pi, Model, LB, \_UB),\\
&\quad UB \geq LB,\\
&\quad upperbound\_constraint\_rev(\sigma, Bt, Model, UB - 1).
\end{aligned}
$$

upperbound_constraint_trans/4

$$
\begin{aligned}
&upperbound\_constraint\_trans(\iota, [\,], \_Model, \_UB).\\
&upperbound\_constraint\_trans(\pi, [B|Bt], Model, UB) :-\\
&\quad transposition\_cop(\pi, \sigma, \_I, \_J, \_K, B),\\
&\quad bound(\pi, Model, LB, \_UB),\\
&\quad UB \geq LB,\\
&\quad upperbound\_constraint\_trans(\sigma, Bt, Model, UB - 1).
\end{aligned}
$$

Especificações Técnicas

Especificações Técnicas:

- Processador: Intel® Core™ 2 Duo 2.33GHz.
- Memória RAM: 3 GB.
- Sistema Operacional: Ubuntu Linux com kernel 2.6.31.
- Compilador para linguagem C++: gcc 4.4.3.
- *ECLiPSe-6.0*.
- *GLPK-4.35*.
- Pacote proprietário para linguagem C++, proprietário *IBM® ILOG® CPLEX® CP Optimizer v 2.3*.
- Pacote proprietário para linguagem C++, proprietário *IBM® ILOG® CPLEX® Optimizer v 12.1*.

Descrição dos Testes

- Para ILP usamos as formulações descritas em Dias e Souza (2007).
- A coluna **size** indica o tamanho das permutações usadas nos testes.
- O tempo de CPU (em segundos) mostrado é o tempo médio usado para resolver 50 permutações aleatórias com tamanho n.
- Fizemos uma comparação entre a permutação π e a permutação identidade ι.
- O tempo limite para resolver uma instância é de 25 horas.
- O caractere "-" significa que o modelo não conseguiu resolver a instância dentro do tempo limite.
- O caractere "*" significa que o modelo não conseguir resolver a instância devido ao limite de memória do sistema.

Resultados: Ordenação por Reversões [1/4]

size	CP-ECLiPSe					
	COP			CSP		
	def	rev_br	rev_cg	def	rev_br	rev_cg
3	0.008	0.003	0.013	0.004	0.004	0.003
4	0.368	0.339	1.490	0.025	0.004	0.003
5	31.676	39.424	93.984	0.531	0.014	0.008
6	-	-	*	9.001	0.032	0.014
7	-	-	-	442.825	0.179	0.072
8	-	-	-	-	1.075	0.449
9	-	-	-	-	5.015	1.401
10	-	-	-	-	51.800	8.639
11	-	-	-	-	1664.691	194.266
12	-	-	-	-	-	671.979
13	-	-	-	-	-	*

Resultados: Ordenação por Reversões [2/4]

size	CP-ILOG					
	COP			CSP		
	def	rev_br	rev_cg	def	rev_br	rev_cg
3	0.001	0.001	0.001	0.001	0.001	0.001
4	0.001	0.001	0.001	0.001	0.001	0.001
5	0.019	0.018	0.019	0.001	0.006	0.004
6	0.144	0.148	0.126	0.001	0.023	0.020
7	1.962	1.807	1.331	0.007	0.156	0.225
8	25.831	26.275	30.244	0.001	0.642	2.757
9	503.421	473.005	405.242	0.003	11.416	54.560
10	-	-	-	*	*	*

Resultados: Ordenação por Reversões [3/4]

size	ILP	
	GLPK	ILOG CPLEX
3	0.001	0.001
4	0.001	0.002
5	0.176	0.039
6	2.118	0.653
7	3.896	21.911
8	3.006	280.528
9	-	258.051
10	-	-

Resultados: Ordenação por Reversões [4/4]

- Não resolveu algumas instâncias devido ao limite de memória do sistema.
- No *ECLiPSe*, os modelos COP, quanto melhor o limitante, maior é o tempo necessário para resolver as instâncias (Complexidade dos limitantes + espaço de busca).
- Nenhum modelo de Ordenação por Reversões conseguiu solucionar as instâncias com permutações de tamanho $n > 13$ dentro do tempo limite de 25 horas.

Resultados: Ordenação por Transposições [1/4]

size	CP-ECLiPSe					
	COP			CSP		
	def	tra_br	tra_cg	def	tra_br	tra_cg
3	0.005	0.006	0.004	0.003	0.002	0.003
4	0.479	1.557	0.049	0.012	0.004	0.003
5	174.445	1275.804	2.922	0.318	0.020	0.005
6	-	-	33.311	15.957	0.185	0.009
7	-	-	-	394.574	0.519	0.015
8	-	-	-	-	7.254	0.069
9	-	-	-	-	99.152	0.365
10	-	-	-	-	-	5.162
11	-	-	-	-	-	33.122
12	-	-	-	-	-	43.893
13	-	-	-	-	-	521.840
14	-	-	-	-	-	1223.512
15	-	-	-	-	-	-

Resultados: Ordenação por Transposições [2/4]

size	CP-ILOG					
	COP			CSP		
	def	tra_br	tra_cg	def	tra_br	tra_cg
3	0.001	0.001	0.001	0.001	0.001	0.001
4	0.004	0.001	0.001	0.001	0.001	0.001
5	0.023	0.009	0.004	0.001	0.008	0.008
6	0.321	0.117	0.072	0.001	0.035	0.041
7	1.747	1.313	0.321	0.005	0.112	0.173
8	29.714	45.806	27.883	0.002	0.651	5.895
9	641.954	838.988	320.615	0.004	2.285	53.673
10	-	-	-	-	-	-

Resultados: Ordenação por Transposições [3/4]

size	ILP	
	GLPK	ILOG CPLEX
3	0.001	0.001
4	0.001	0.005
5	0.376	0.054
6	3.948	2.308
7	5.558	66.497
8	-	-

Resultados: Ordenação por Transposições [4/4]

- Os modelos que usam o limitantes **tra_br_cop** apresentaram os piores resultados.
- No *ECLiPSe*, os modelos COP, o melhor limitante possui o melhor tempo de execução.
- Nenhum modelo de Ordenação por Transposições conseguiu solucionar as instâncias com permutações de tamanho $n > 14$ dentro do tempo limite de 25 horas.

Resultados: Ordenação por Reversões e Transposições [1/4]

size	CP-ECLiPSe							
	COP				CSP			
	def	r_t_br	r_t_bc	r_t_cc	def	r_t_br	r_t_bc	r_t_cc
3	0.035	0.010	0.004	0.006	0.004	0.005	0.004	0.008
4	6.434	10.333	0.248	0.687	0.022	0.026	0.035	0.081
5	-	-	30.824	64.170	0.379	0.414	0.680	2.066
6	-	-	519.803	-	11.566	12.776	21.264	89.760
7	-	-	-	-	400.904	441.946	792.422	*
8	-	-	-	-	-	-	-	-

Resultados: Ordenação por Reversões e Transposições [2/4]

size	CP-ILOG							
	COP				CSP			
	def	r_t_br	r_t_bc	r_t_cc	def	r_t_br	r_t_bc	r_t_cc
3	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
4	0.002	0.002	0.002	0.003	0.001	0.001	0.001	0.001
5	0.019	0.020	0.022	0.023	0.001	0.001	0.001	0.001
6	0.340	0.327	0.319	0.319	0.001	0.001	0.001	0.001
7	2.052	2.062	2.066	2.088	0.004	0.004	0.002	0.004
8	11.894	11.727	12.367	12.369	0.002	0.004	0.004	0.004
9	310.012	304.331	331.275	328.956	0.006	0.006	0.004	0.005
10	-	-	-	-	0.012	0.010	0.010	0.011
11	-	-	-	-	-	-	-	-

Resultados: Ordenação por Reversões e Transposições [3/4]

size	ILP	
	GLPK	ILOG CPLEX
3	0.001	0.001
4	0.008	0.008
5	0.466	0.059
6	4.642	1.374
7	5.094	82.241
8	-	-

Resultados: Ordenação por Reversões e Transposições [4/4]

- Obteve os piores resultados. Espaço de busca maior por realizar as duas operações em conjunto.
- No *ECLiPSe*, no modelo CSP que usa o limitante **r_t_cc_csp**, não foi possível resolver a instância com permutações de tamanho $n = 7$ devido ao limite de memória do sistema.
- Rápido crescimento do espaço de busca conforme o tamanho das permutações.
- Nenhum modelo de Ordenação por Reversões conseguiu solucionar as instâncias com permutações de tamanho $n > 10$ dentro do tempo limite de 25 horas.

Resultados [1/2]

- Os modelos COP possuem o pior tempo de execução.
- Busca uma solução base para depois encontrar uma solução melhor, usando uma função de custo.
- Espaço de busca é maior em relação aos modelos CSP.
- Diferença na complexidade dos modelos desenvolvidos para ILOG. Motivos:
  - Forma de modelagem. Modelo foi pensado para o *ECLiPSe*.
  - *Backtracking*. O *ECLiPSe* é baseado no prolog (paradigma de programação lógica) e possui o mecanismo embutido na linguagem. Já os modelos para o ILOG foram escrito na linguagem C++.

Resultados [2/2]

Em ILP:

- ILOG CPLEX foi melhor nas permutações com $n < 7$.
- GLPK foi melhor nas permutações com $n = 7$, mas não conseguiu resolver permutações com $n > 8$.

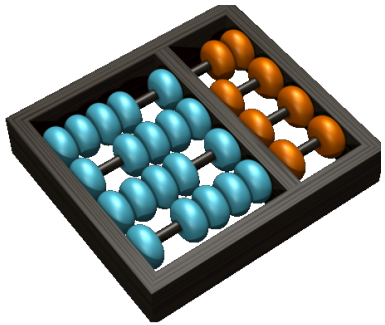
Em CP:

- Teoria COP: ILOG CP foi melhor nos três casos.
- Teoria CSP: *ECLiPSe* obteve tempos melhores e resolveu instâncias maiores, exceto no caso de ordenação por reversões e transposições.

Conclusões e Trabalhos Futuros

- Resultados mostraram que os modelos de CP baseados na teoria CSP obtiveram os melhores tempos em relação aos modelos de ILP.
- Em ordenação por reversões, os modelos de CP baseados na teoria COP obtiveram os piores resultados. As formulações ILP obtiveram os piores resultados nos outros problemas.
- Esta abordagem ainda é inviável na prática.
- Heurísticas podem ser estudadas para melhorar o desempenho dos modelos de CP (Redução do espaço de busca).
- Para ILP, melhorar a formulação usando técnicas como relaxação lagrangeana, ou escrever uma nova formulação, sem se preocupar com o seu tamanho, para aplicar técnicas de geração de colunas e *branch-and-cut*.

Agradecimentos



UNICAMP

CNPq
Conselho Nacional de Desenvolvimento
Científico e Tecnológico