

# Reduções

André Vignatti

DINF- UFPR

O que significa **transformar** (reduzir) um problema  $A$  em um problema  $B$ ?

- 1 Pegar a entrada (instância) de  $A$  e transformar numa entrada de  $B$ .
- 2 Usar o algoritmo de  $B$  como **caixa-preta**.
- 3 Transformar a saída (resposta) de  $B$  numa saída de  $A$ .

Reduzir um problema  $A$  para um problema  $B$ :

O algoritmo para  $A$  usa o algoritmo para  $B$  como **caixa-preta**.

# Redução 0 - Problema dos Elementos Distintos

Dado um vetor  $A$  de  $n$  inteiros, há alguma **duplicata** em  $A$ ?

Algoritmo Natural:

---

```
1  para  $i \leftarrow 1$  até  $n - 1$  faça  
2      para  $j \leftarrow i + 1$  até  $n$  faça  
3          se  $A[i] = A[j]$   
4              retorne SIM  
5  retorne NÃO
```

---

Tempo de execução:  $O(n^2)$ .

# Redução 0 - Problema dos Elementos Distintos

Redução para Ordenação:

---

```
1  Ordene A
2  para  $i \leftarrow 1$  até  $n - 1$  faça
3      se  $A[i] = A[i + 1]$ 
4          retorne SIM
5  retorne NÃO
```

---

Tempo de execução:  $O(n)$  mais o tempo de ordenar  $n$  números.

O algoritmo usa a ordenação como caixa-preta.

# Os Dois Lados da Redução

Suponha que um problema  $A$  se reduz a um problema  $B$ :

**Direção Positiva** : o algoritmo para  $B$  implica num algoritmo para  $A$ .

**Direção Negativa** : suponha que **não** há algoritmo eficiente para  $A$ . Então **não** há algoritmo eficiente para  $B$  (uma restrição técnica do tempo de redução é necessário neste caso)

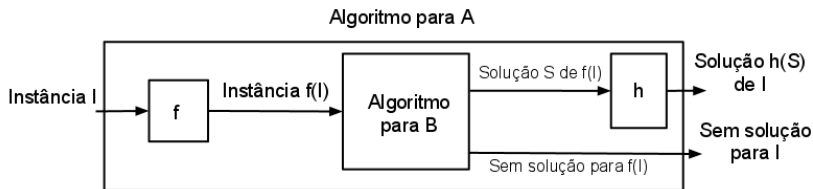
## Exemplo:

**Direção Positiva** : um algoritmo  $O(n \log n)$  para a **Ordenação** implica num algoritmo  $O(n \log n)$  para os **Elementos Distintos**.

**Direção Negativa** : se não há algoritmo melhor que  $n \log n$  para os **Elementos Distintos** então não há algoritmo melhor que  $n \log n$  para a **Ordenação**.

(A **direção negativa** será vista com mais calma na redução do Ordenação para o Casco Convexo) .

# Redução entre problemas - Esquema



Claramente vale transitividade:

Se  $P_1 \rightarrow P_2$  e  $P_2 \rightarrow P_3$  então  $P_1 \rightarrow P_3$

## Problema do Casamento Cíclico - CC

Sejam  $A$  e  $B$  duas strings de tamanho  $n$  onde:

$$A = a_0 a_1 \dots a_{n-1} \quad \text{e} \quad B = b_0 b_1 \dots b_{n-1}$$

Determinar se  $B$  é uma rotação cíclica de  $A$ .

Isto é, determinar se existe índice  $k$  tal que:

$$a_{[(i+k) \bmod n]} = b_i, \quad \text{para } i = 0, \dots, n-1.$$

ABCD é uma rotação cíclica de CDAB com  $k = 2$

Podemos resolver este problema diretamente, mas vamos resolvê-lo reduzindo-o a outro.



## Problema do Casamento de String - CS

Sejam  $S$  (string) e  $P$  (padrão) duas strings. Determinar se  $P$  é uma substring de  $S$ .

## Proposicao

Casamento Cíclico  $\rightarrow$  Casamento de String.

## Demonstração.

Se  $(A, B)$  é instância do problema CC então

$B$  é uma rotação cíclica de  $A \Leftrightarrow B$  é uma substring de  $AA$ .

Assim,

$(A, B)$  de CC tem resposta sim  $\Leftrightarrow (AA, B)$  de CS tem resposta sim



Exercicio: Caso um algoritmo de CS devolva o índice onde começa o padrão, faça uma redução que devolva também a posição da rotação cíclica.

## Definicao

Dada coleção de conjuntos  $S_1, \dots, S_k$  temos que  $R = \{r_1, \dots, r_k\}$  é um Sistema de Representantes Distintos (SRD) se  $r_i \in S_i$  para todo  $i = 1, \dots, k$ .

## Problema do Sistema de Representantes Distintos

Dado coleção de conjuntos  $S_1, \dots, S_k$ , encontrar um SRD.

## Exemplo:

Considere os seguintes conjuntos:

*Ecológicos*: Ana, Alberto

*Ruralistas*: João, Alberto

*Feministas*: Ana, Maria

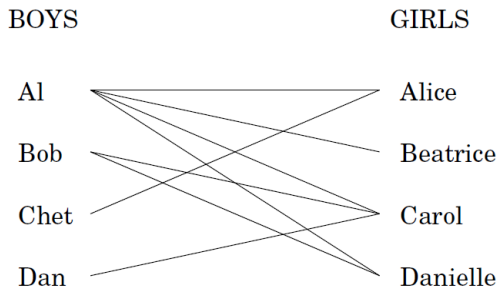
Então, {Ana, João, Maria} formam um SRD.

Não há SRD para a coleção abaixo:

- $S_1 = \{1, 2\}$
- $S_2 = \{3, 4\}$
- $S_3 = \{3, 4\}$
- $S_4 = \{1, 2, 4\}$
- $S_5 = \{2, 4\}$

## Problema do Emparelhamento Máximo - EM

Dado grafo bipartido  $G = (X, Y, E)$ , onde  $X$  e  $Y$  são conjuntos de vértices e  $E$  é o conjunto de arestas, encontrar um emparelhamento (conjunto de arestas sem extremos em comum) de cardinalidade máxima.



## Proposição

Problema do SRD  $\rightarrow$  Problema do EM.

## Demonstração.

Dada coleção  $S_1, \dots, S_k$ , instância de um SRD, sobre conjunto  $A$ ,

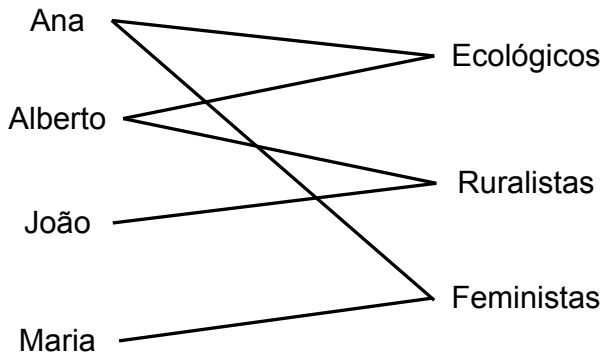
defina grafo  $G = (X, Y, E)$  onde

- $X = S_1 \cup S_2 \cup \dots \cup S_k$
- $Y = \{1, 2, \dots, k\}$
- $E = \{(a, j) \mid a \in S_j\}$ .

Note que o SRD tem solução  $\Leftrightarrow G$  tem emparelhamento de tamanho  $k$ . (Seria necessário uma prova mais formal desse fato)  $\square$

# Redução 2

Para o exemplo de antes:



# Redução 3

## Problema do Triângulo em um Grafo

Dado grafo  $G = (V, E)$  não-orientado, decidir se  $G$  tem um triângulo (subgrafo completo de 3 vértices).

## Proposicao

O Problema dos Triângulo em um grafo pode ser resolvido em tempo  $O(n^3)$ .

## Demonstração.

Basta verificar todos os  $\binom{n}{3}$  subconjuntos de 3 vértices de  $G$ . □

# Redução 3

Vamos reduzir o Problema do Triângulo em um Grafo para o seguinte problema

## Problema da Multiplicação de Matrizes

Dadas matrizes  $A$  e  $B$ , computar a matriz  $C = A \cdot B$ .

## Teorema [Strassen'69]

Existe algoritmo que computa multiplicação de matrizes de ordem  $n$  com complexidade de tempo  $O(n^{2.807})$ .

## Teorema [Coppersmith-Winograd'90]

Existe algoritmo que computa multiplicação de matrizes de ordem  $n$  com complexidade de tempo  $O(n^{2.376})$ .



## Definicao

Dado grafo  $G = (V, E)$ , onde  $V = \{1, \dots, n\}$ , a matriz de adjacência  $A$  de  $G$  é uma matriz de ordem  $n \times n$  onde

$$A[i, j] = \begin{cases} 1 & \text{se } \{i, j\} \in E \\ 0 & \text{caso contrário.} \end{cases}$$

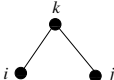
O que acontece se calcularmos  $A^2$  ?

Sabemos que

$$A^2[i, j] = \sum_{k=1}^n A[i, k] \cdot A[k, j]$$

Portanto  $A^2[i, j] > 0 \Leftrightarrow$  existe índice  $k$  tal que  $A[i, k] = 1$  e  $A[k, j] = 1$ .

I.e.,

$$A^2[i, j] > 0 \Leftrightarrow \text{existe } k \text{ tal que}$$


## Proposicao

O Problema do Triângulo em um grafo  $G$  pode ser resolvido em tempo  $O(n^{2^{376}})$ .

## Demonstração.

Seja  $A$  matriz de incidência de  $G$ .

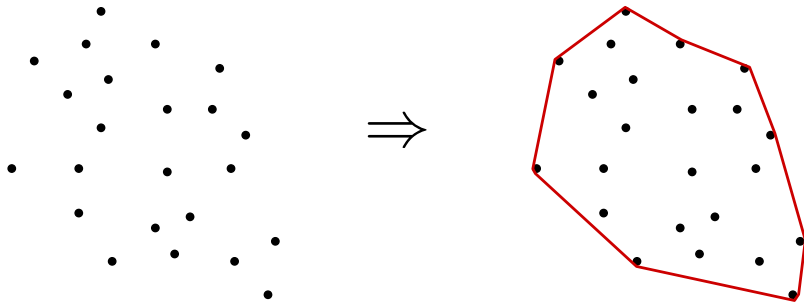
- Compute  $A^2$  em tempo  $O(n^{2^{376}})$ .
- Para cada posição  $(i, j)$ , verifique em  $A$  e  $A^2$  se  $A[i, j] = 1$  e  $A^2[i, j] > 0$ .
- Caso se verifique para algum par, temos um triângulo em  $G$ .



# Redução 4

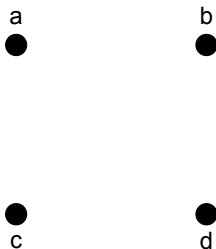
## Problema do Casco Convexo

São dados  $n$  pontos no plano  $p_1, \dots, p_n$ . Queremos os pontos do casco convexo listados em sentido anti-horário.



## Redução 4

**Não basta** retornar os pontos do casco convexo: deve-se **também retorná-los em sentido anti-horário**.



- $(a, c, d, b)$  é uma solução válida.
- $(d, b, a, c)$  é uma solução válida.
- $(a, d, c, b)$  **não** é uma solução válida.

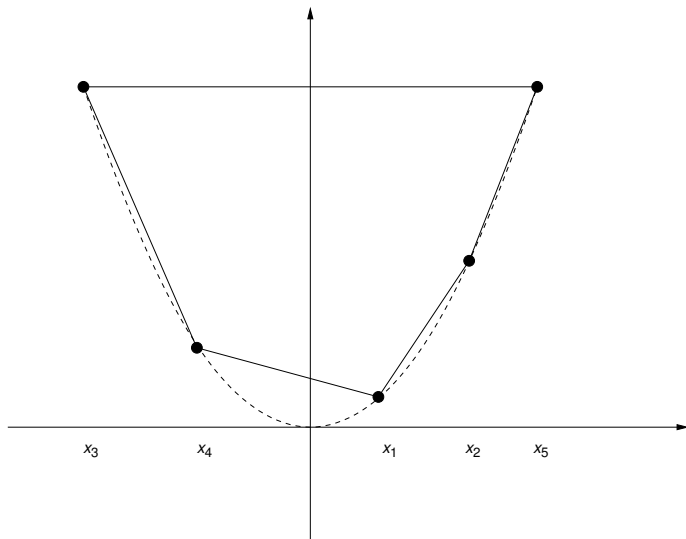
Iremos reduzir o problema da **ordenação** no problema do **casco convexo**.

Ou seja, vamos usar um algoritmo do casco convexo como caixa-preta para resolver a ordenação.

## Redução:

- 1 Sejam  $x_1, x_2, \dots, x_n$  os números da entrada do problema da ordenação.
- 2 Para cada  $x_i$ , transforme num ponto 2D  $(x_i, x_i^2)$  (esses pontos estão na parábola  $y = x^2$ ).
- 3 Execute o algoritmo do casco convexo (claramente os pontos estão no casco convexo, resta saber a ordem).
- 4 Encontre o menor ponto (leva  $O(n)$ ) e a partir dele, percorra a ordem retornada.

# Redução 4





Os pontos retornados pelo casco convexo (começando do menor) são os elementos ordenados.

Então, podemos usar o algoritmo do casco convexo para ordenar!

De um certo ponto de vista...

...o problema do casco convexo é **mais difícil** que o problema da ordenação (pois ele resolve a ordenação).

## Conclusão:

Se tivermos um algoritmo rápido para o casco convexo, podemos usar para ordenar.

- Mas sabemos que para ordenar é necessário pelo algoritmo  $\Omega(n \log n)$ .
- Então, **não existe algoritmo** para o **casco convexo** melhor que  $\Omega(n \log n)$ .