Projeto e Análise de Algoritmos

Prof. Hamilton José Brumatto

Bacharelado em Ciência da Computação - UESC

11 de novembro de 2010

Reduções

Slides construídos com base nas notas de aula do prof.
Pedro Jussieu de Rezende

Modelo Computacional

- A construção de algoritmos implica na definição de um conjunto de passos para se resolver um problema.
- A análise da complexidade do algoritmo envolve avaliar o custo computacional das operações em cada passo.
- O custo de cada operação pode mudar de acordo com o modelo computacional adotado.
- A complexidade do algoritmo depende então do modelo computacional em que é implementado.
- Só faz sentido comparar complexidade de algoritmos distintos para um mesmo problema, se considerarmos a implementação em um modelo computacional comum.
- Da mesma forma, a definição do modelo computacional para análise de um problema é essencial para avaliar a dificuldade intrínsica do Problema (Cota inferior do problema).

Cota Inferior e Cota Superior

Cota Superior

- Seja um problema, se conhecemos um algoritmo que resolva este problema em um modelo computacional, e esta algoritmo possui a complexidade de tempo expressa por $T(n)$, então dizemos que $T(n)$ é uma Cota Superior para o problema.
- Observe que pode existir um algoritmo pior, no entanto sabemos que o problema pode ser resolvido em $T(n)$, logo os algoritmos que buscamos precisam ser melhores na complexidade de tempo do que $T(n)$.

Cota Inferior e Cota Superior

Cota Inferior

- Seja um problema, e um modelo computacional no qual acredita-se que é possível resolver P (mesmo que não exista ainda algoritmo que o resolva).
- Seja $I(n)$ uma função que expressa a eficiência mínima para qualquer algoritmo neste modelo computacional que possa resolver o problema, então dizemos que $I(n)$ é uma Cota Inferior para o problema.

Exemplo de Cota Inferior

- No início do curso adotamos o modelo RAM (*Random Access Machine*) como modelo computacional*.
- Neste modelo trabalhamos alguns algoritmos de Ordenação, e a melhor complexidade obtida foi $\Theta(n \log n)$.
- Os algoritmos se baseiam na comparação entre dois valores que resultam em duas soluções: MENOR ou NÃO MENOR.
- Podemos construir uma árvore binária onde cada nó é uma comparação e o filho de cada nó um dos possíveis resultados, a partir de onde faremos novas comparações.
- Uma das folhas será a solução, e a árvore de menor altura apresenta o menor caminho para a solução.

*Como exemplo de um modelo diferente: Computação Quântica

Árvores de Decisão para o problema de ordenação

- Quantas comparações existe em uma ordenação?
  - A ordenação é baseada em troca, portanto, se temos n elementos, existe uma permutação P_n possíveis ordens entre os elementos (não consideramos elementos repetidos).
  - Mas, $P_n = n!$.
  - Sabemos que uma árvore binária com k folhas tem altura mínima $\log k$[†].
  - A altura mínima de nossa árvore de decisão é de $\log n!$.
- Isto significa que a solução não pode ser melhor que $\Omega(\log n!) = \Omega(n \log n)$.
- Os algoritmos que encontramos $\Theta(n \log n)$ são ótimos, neste modelo (e sem impor restrições ao problema) não dá para fazer melhor.

[†]Fica como exercício provar isto

Reduções

- Mas é simples encontrar a cota inferior de um problema? NÃO
- Uma possível forma seria a de comparar problemas.
- Vamos considerar dois problemas P, e Q.
- O conjunto de Instâncias de P é indicado por: $\mathbb{I}_P$, e o conjunto de instâncias de Q é indicado por: $\mathbb{I}_Q$.
- O conjunto de Soluções de P é indicado por: $\mathbb{S}_P$, e o conjunto de Soluções de Q é indicado por $\mathbb{S}_Q$.
- Um algoritmo $\mathcal{A}$ que resolve Q consegue transformar cada instância $i \in \mathbb{I}_Q$ em uma solução $s \in \mathbb{S}_Q$.

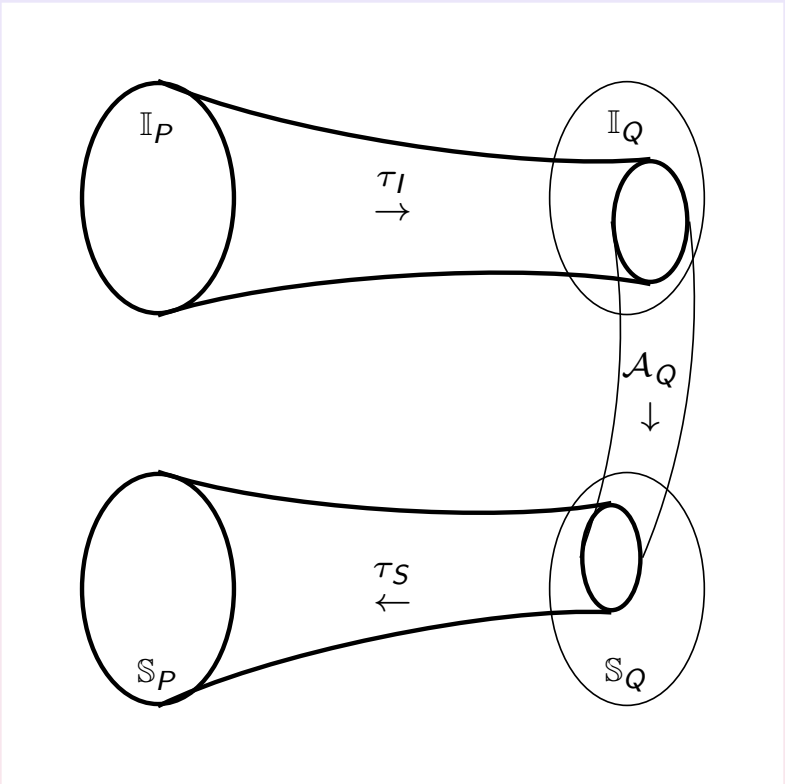
Reduzindo problemas

- Vamos considerar duas funções: τ_I injetora e τ_S sobrejetora, definidas como:

$$\begin{aligned} \tau_I &: \mathbb{I}_P \rightarrow \mathbb{I}_Q \\ \tau_S &: \mathbb{S}_Q \rightarrow \mathbb{S}_P \end{aligned}$$

- Assim, se $\mathbb{S}_Q$ é solução de $\tau_I(\mathbb{I}_P)$, então $\tau_S(\mathbb{S}_Q)$ é solução de $\mathbb{I}_P$.
- Se existe o algoritmo $\mathcal{A}_Q$ que resolve Q, então podemos definir um algoritmo $\mathcal{A}_P = \tau_I \circ \mathcal{A}_Q \circ \tau_S$.

Reduzindo problemas



$$\mathcal{A}_P(\mathbb{I}_P) = \tau_S(\mathcal{A}_Q(\tau_I(\mathbb{I}_P)))$$

Complexidade de Tempo nas Reduções

- Seja n o tamanho das entradas de P e de Q. Sejam $T_{\tau_I}(n)$ e $T_{\tau_S}(n)$ funções que representam o comportamento assintótico da complexidade das transformações τ_I e τ_S.
- Considere que: $T_{\tau_I}(n) + T_{\tau_S}(n) \in O(f(n))$.
- Dizemos que P é redutível em tempo $f(n)$ a Q e denotamos este fato por

$$P \propto_{f(n)} Q$$

- Se Q tem quota superior $O(g(n))$ e $P \propto_{f(n)} Q$, então P tem quota superior $O(g(n) + f(n))$.
- Se P tem quota inferior $\Omega(h(n))$ e se $P \propto_{f(n)} Q$ e $f(n) \in o(h(n))$, então Q tem quota inferior $\Omega(h(n))$[‡].

[‡]Prove isto!

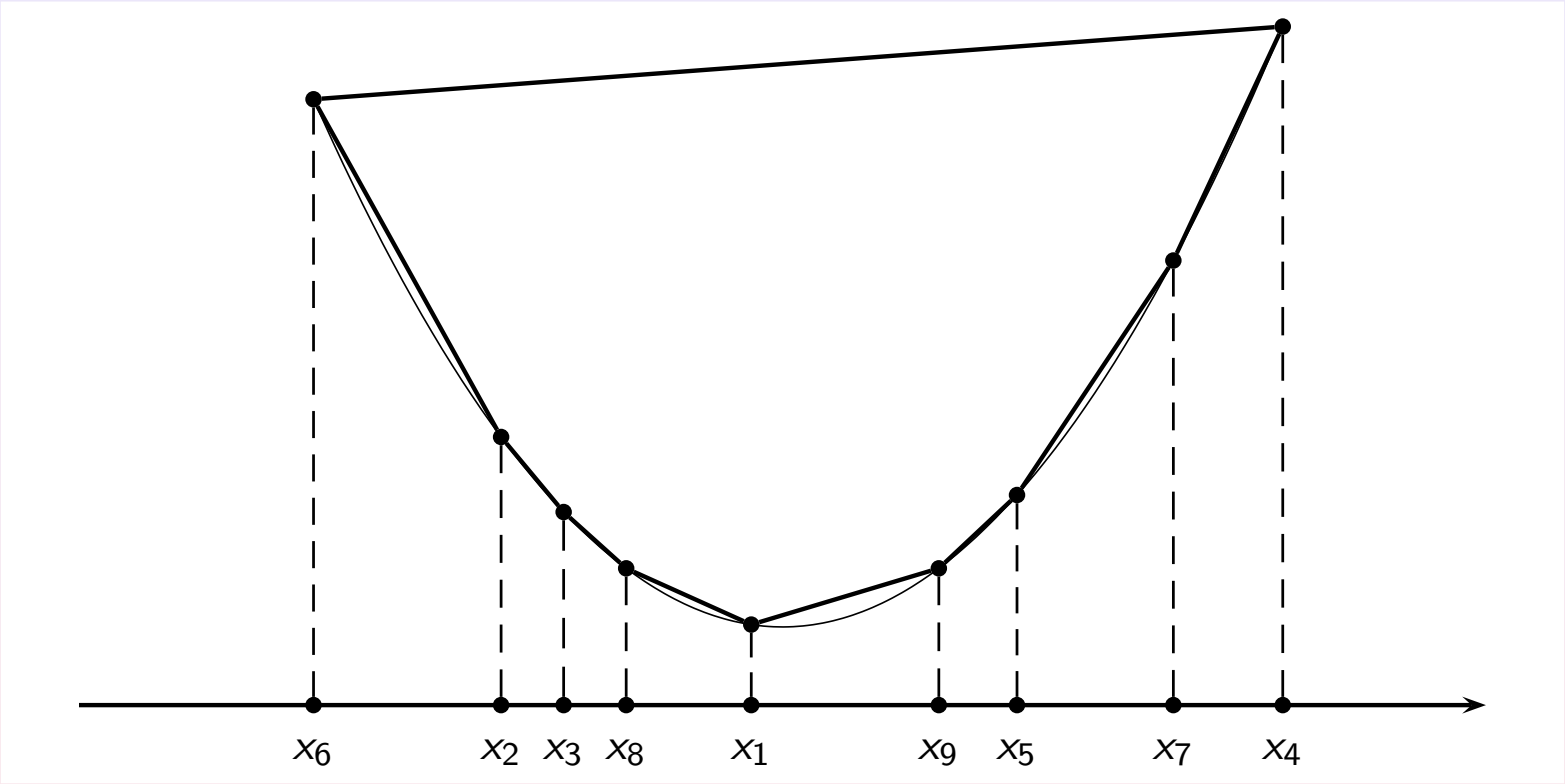
Problema da Envoltória Convexa (EC)

- Qual a cota inferior para o problema da envoltória convexa (EC)? (Problema Q)
- Sabemos a cota inferior para o problema de ordenação (Ord): $\Omega(n \log n)$ (Problema P).
- Se conseguirmos transformar cada instância do problema de ordenação (Ord) em uma instância do problema da envoltória convexa, e após resolver este problema, conseguirmos transformar a solução do problema da envoltória convexa em uma solução do problema de ordenação, tudo isto em tempo $f(n) = o(n \log n)$, então teremos uma cota inferior para o problema da Envoltória Convexa (EC).
- Resumindo: Se conseguirmos $Ord \propto_{f(n)} EC$, com $f(n) \in o(n \log n)$, então a cota inferior do problema da envoltória convexa é $\Omega(n \log n)$.

Reduzindo o problema da Ordenação para o problema da Envoltória Convexa

- Considere: $\tau_I : \mathbb{I}_{Ord} \rightarrow \mathbb{I}_{EC}$ da seguinte forma:
  - Dada uma instância $I_{Ord} = (x_1, x_2, \ldots, x_n)$, construir o conjunto de pontos para uma instância $I_{EC} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, onde $y_i = x_i^2$, para $1 \leqslant i \leqslant n$.
- Claramente $\tau_I \in \Theta(n)$.
- Considere: τ_S da seguinte forma: Buscar no conjunto de vértices retornados, o vértice de menor abscissa. A partir deste ponto, retornar seqüencialmente as abscissas dos vértices que forma a envoltória convexa: $\tau_S \in \Theta(n)$.
- Logo, $f(n) = n$, $(\tau_I + \tau_S) \in O(f(n)$ e $f(n) \in o(n \log n)$.
- O problema da Envoltória convexa tem cota inferior $\Omega(n \log n)$.

Exemplo da redução



Envoltória Convexa ordenando os valores das abscissas

Problemas de Decisão

- Alguns problemas possuem como solução apenas a resposta: SIM ou NÃO.
- Não é possível fazer a redução de um problema que não é de decisão para um problema de decisão (o inverso é possível).
- Para conhecer uma cota inferior de um problema desta classe, primeiro é preciso conhecer a conta inferior de algum problema de decisão para que possamos tentar a redução.

Problema da Unicidade de Elementos (UE)

O Problema

Dada uma coleção de n objetos de um domínio, determinar se eles são todos distintos.

- Não vamos provar aqui, a cota inferior deste problema é $\Omega(n \log n)$.
- Um algoritmo muito simples resolve este problema em $\Theta(n \log n)$

Problema da Unicidade de Elementos (UE)

O Problema

Dada uma coleção de n objetos de um domínio, determinar se eles são todos distintos.

- Não vamos provar aqui, a cota inferior deste problema é $\Omega(n \log n)$.
- Um algoritmo muito simples resolve este problema em $\Theta(n \log n)$

- Ordene os valores: $\Theta(n \log n)$ e faça uma busca de par idêntico consecutivo. $O(n)$.
- Este não seria um exemplo de redução para o problema da Unicidade de Elementos?

Problema da Unicidade de Elementos (UE)

O Problema

Dada uma coleção de n objetos de um domínio, determinar se eles são todos distintos.

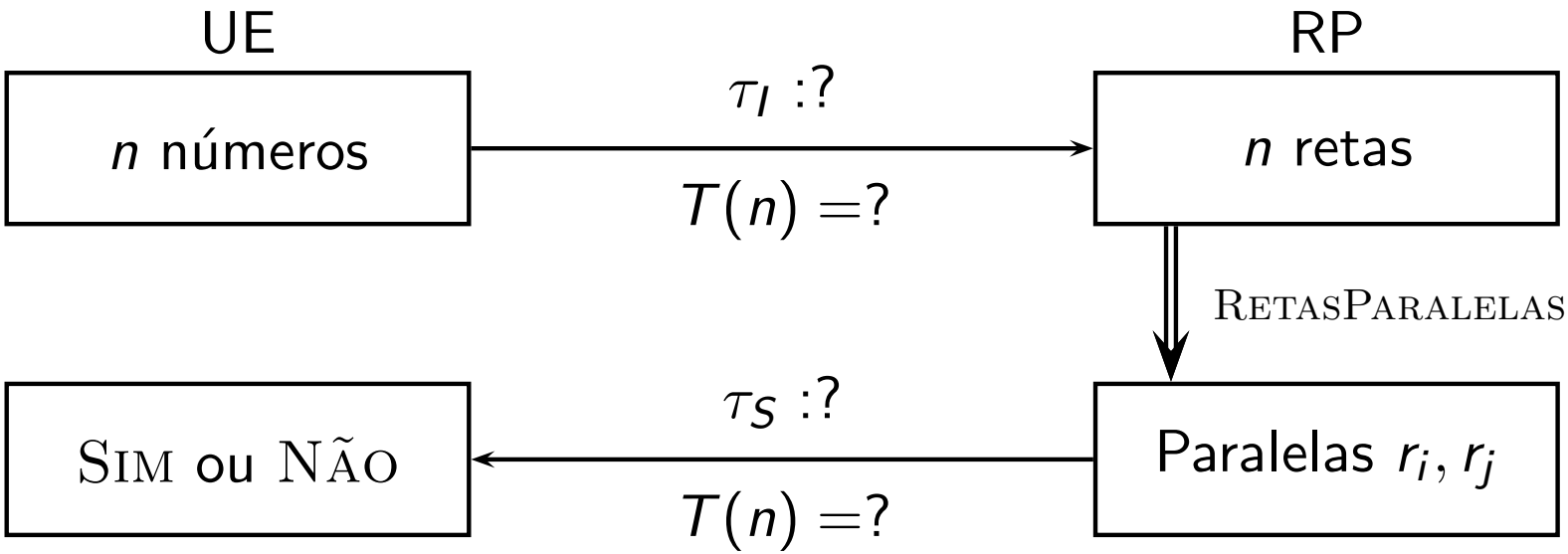
- Não vamos provar aqui, a cota inferior deste problema é $\Omega(n \log n)$.
- Um algoritmo muito simples resolve este problema em $\Theta(n \log n)$

- Ordene os valores: $\Theta(n \log n)$ e faça uma busca de par idêntico consecutivo. $O(n)$.
- Este não seria um exemplo de redução para o problema da Unicidade de Elementos?

Redução: Decisão $\Longrightarrow$ Não Decisão

Problema das Retas Paralelas (RP)

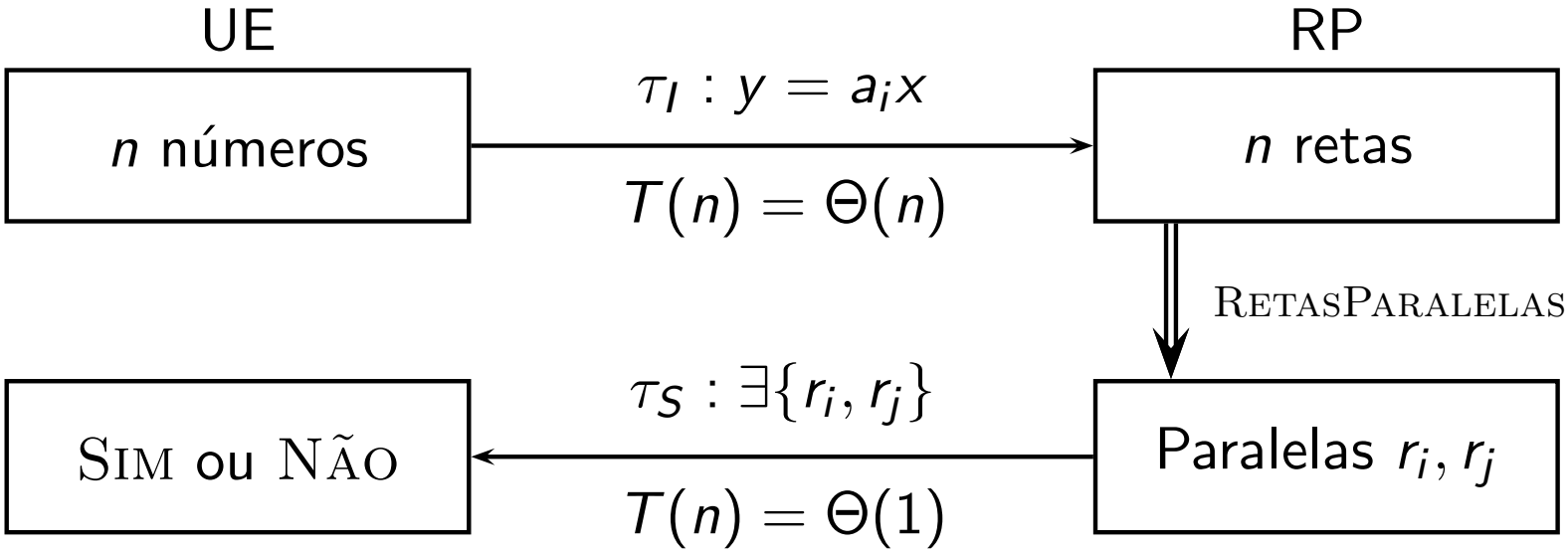
Dada uma coleção de n retas no plano, determinar se há duas delas paralelas



Redução: Decisão $\Longrightarrow$ Não Decisão

Problema das Retas Paralelas (RP)

Dada uma coleção de n retas no plano, determinar se há duas delas paralelas



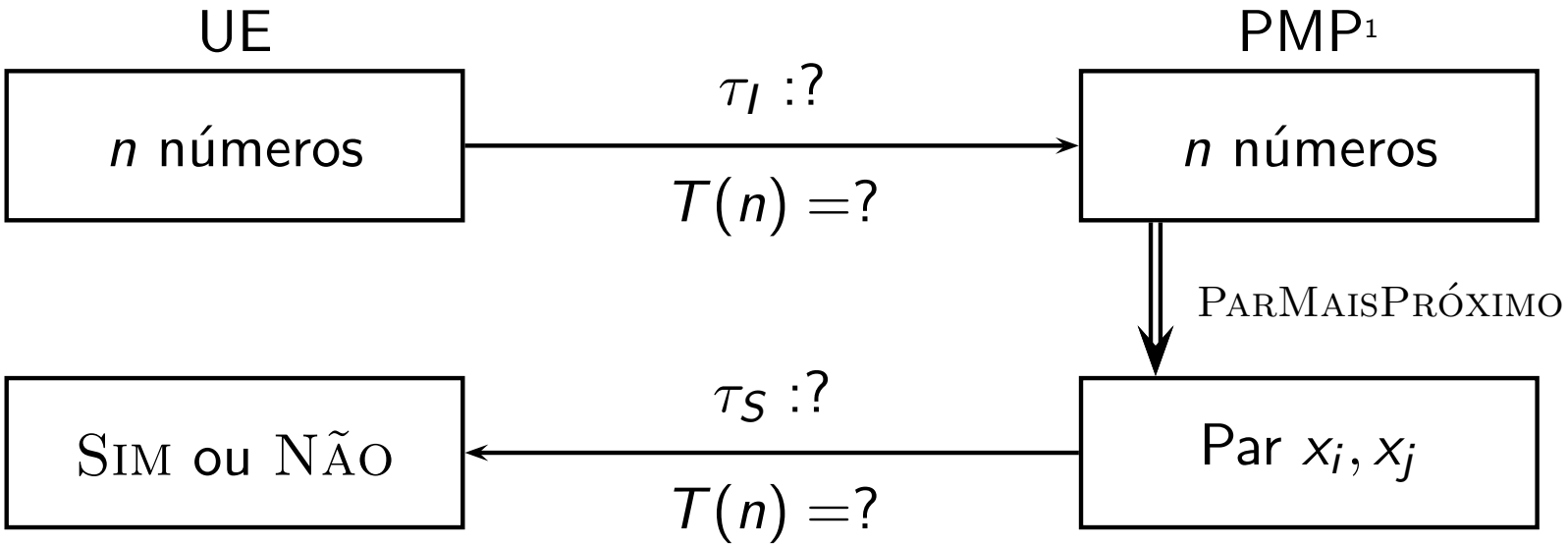
Conclusão: RP tem cota inferior $\Omega(n \log n)$

Redução: Decisão $\Longrightarrow$ Não Decisão

Problema do Par mais Próximo de uma dimensão (PMP^1)

Dada uma coleção de valores algébricos $(x_1, x_2, \ldots, x_n)$, determinar

$$\min\{|x_i - x_j| : 1 \leqslant i, j \leqslant n \text{ e } i \neq j\}$$

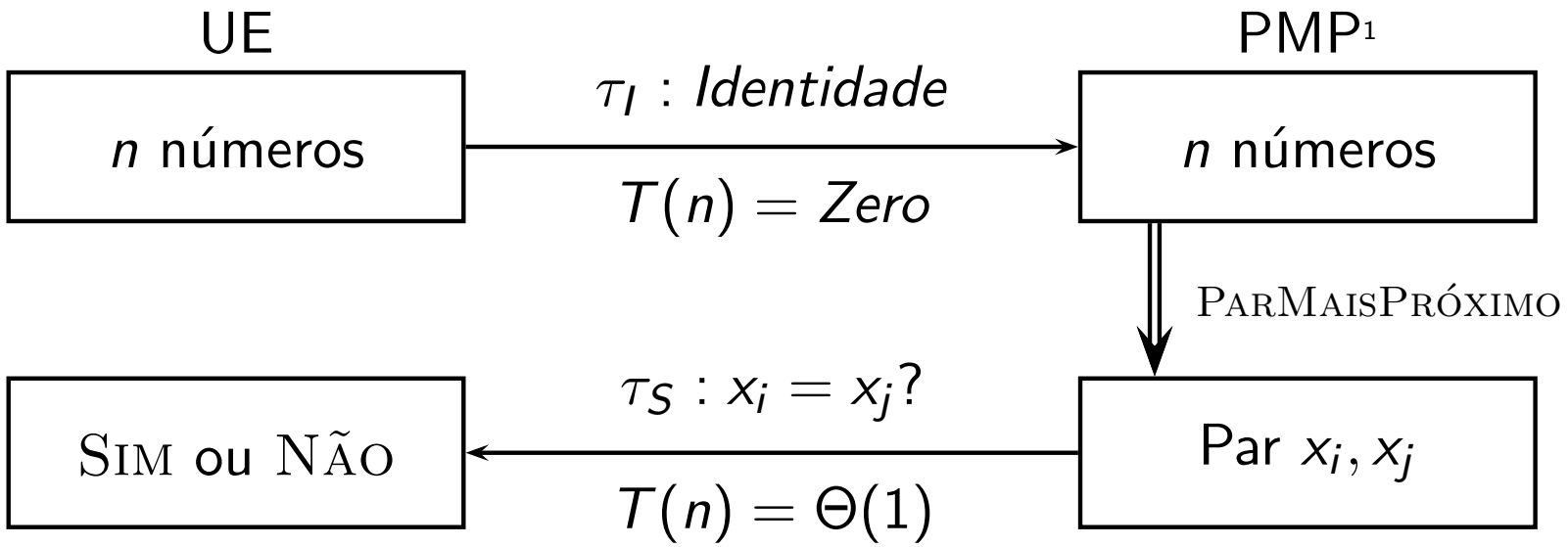


Redução: Decisão $\Longrightarrow$ Não Decisão

Problema do Par mais Próximo de uma dimensão (PMP^1)

Dada uma coleção de valores algébricos $(x_1, x_2, \ldots, x_n)$, determinar

$$\min\{|x_i - x_j| : 1 \leqslant i, j \leqslant n \text{ e } i \neq j\}$$



Conclusão: PMP^1 tem cota inferior $\Omega(n \log n)$

Atividades construídas pelo prof. Rezende

1. Resolver a Lista de Exercícios indicada no moodle para este tópico.