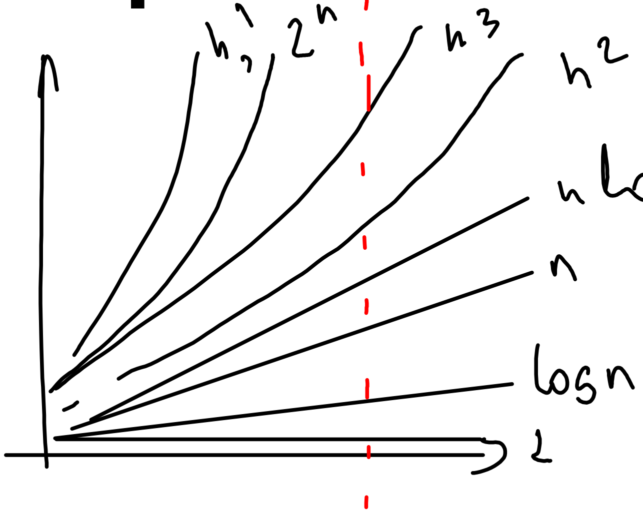
Complexidade Computacional



Bubble Sort $\in P$

Merge Sort $\in P$

busca binária $\in P$

tratável

Classe P

- Algoritmo viável na prática
  - Executam um número de passos limitado por um **polinômio** sobre o comprimento da entrada
- Definição:
  - Uma máquina de Turing é polinomialmente limitada se há um polinômio p(n), tal que a máquina sempre pare após, no máximo, p(n) passos, onde n é o comprimento da entrada.

$O(n^3)$

$P(n) \longrightarrow n^3 + n^2 + 35$

Classe P

- Linguagem polinomialmente decidível
  - Se houver uma máquina de Turing polinomialmente limitada que a decida
  - A classe de todas as linguagens polinomialmente decidíveis é chamada de **Classe P**

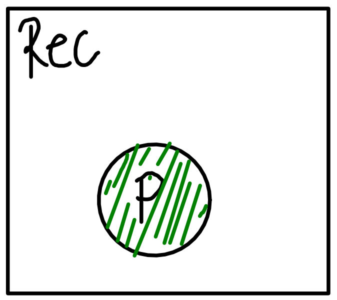
  - Classe P: representam problemas solúveis na prática

Classe P

- Limitados polinomialmtente: O(p(n))
  - Ordenação
    - Merge Sort, Heap Sort: p(n) = n.log n
  - Busca
    - Binária: p(n) = log n
  - Reconhecimento de padrões
  - multiplicação de matrizes
  - FFT, etc.

Não
RE
X

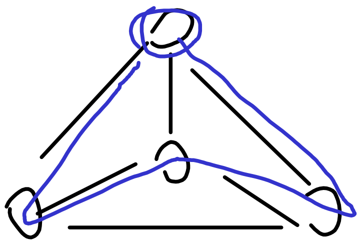
RE
X



Problemas

$A\cdot\bar{B}\cdot C + AB\bar{C} + \bar{A}B\bar{C}$

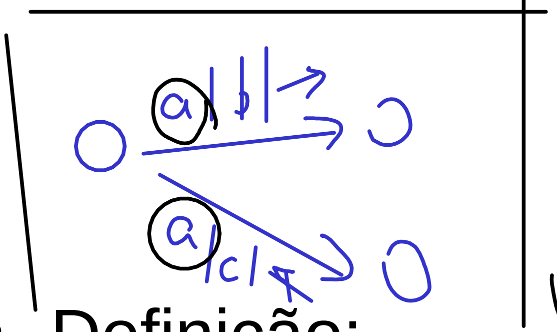
- Problemas naturais, de interesse prático, não pertencem a Classe P
  - Circuito de Hamilton → $\notin$? mas $\in$ NP
  - Caixeiro Viajante
  - Satisfatibilidade
  - Coloração de Grafos → $\notin$ P mas $\in$ NP



- Nenhum algoritmo de resposta polinomial foi desenvolvido até hoje

Problemas

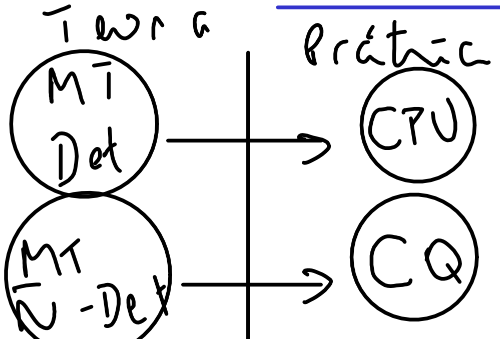
- Solução:
  - Provar que algoritmos de resposta polinomial para estes problemas não são possíveis?
- Infelizmente não!
  - Alguns desses problemas podem ser resolvidos por máquinas de Turing não-determinísticas polinomialmente limitadas



Classe NP

~~Não Polinomial~~

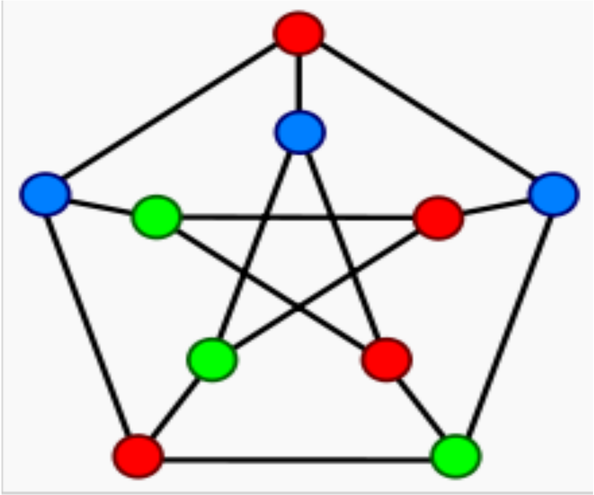
- Definição: não determinismo
  - Uma máquina de Turing não-determinística é polinomialmente limitada se há um polinômio p(n), tal que nenhuma computação nessa máquina dure mais que p(n) passos, onde n é o comprimento da entrada.
  - Todas as linguagens decididas por máquinas de Turing não-determinísticas, limitadas polinomialmente, formam a Classe NP

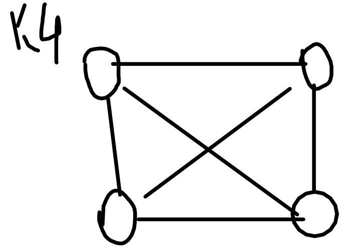


Exemplo

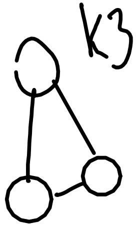
- Coloração de Grafos:
  - uma forma de colorir os vértices de um grafo tal que não haja dois vértices adjacentes que compartilhem a mesma cor

$K = 3$





n



K1&2

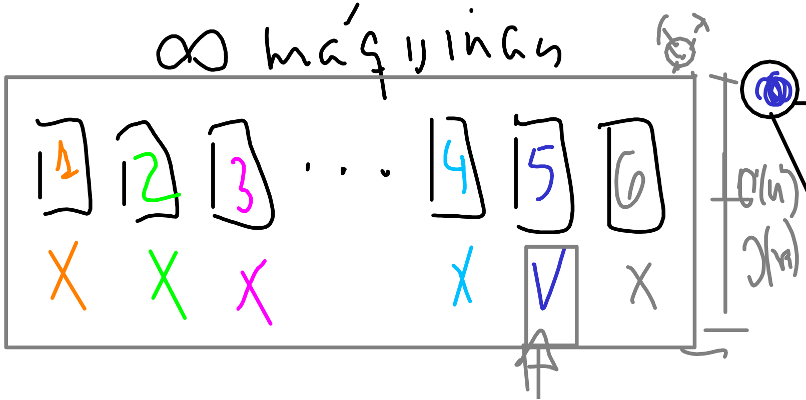
$n(n^2-1)(n^2-n) \to O(n^2)$

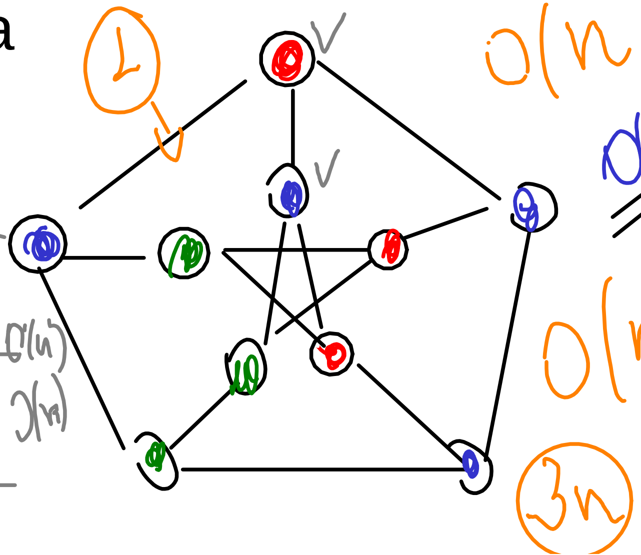
Exemplo

Não Det Polinomial

- um grafo é dito ser **k-colorível** se ele tem uma coloração própria de vértices que usa k cores
  - Decidir se um grafo é 3-colorável por exemplo é um problema NP, pois nenhum algoritmo de tempo polinomial pode obter a coloração mínima
- No entanto...

∞ máquinas





$O(n)$

$O(p(n))$

$O(n^2)$

1ª Fase "Chute"

Não Det

2ª Fase "Validar"

Det

o Chute

$O(n)$

Exemplo

- Dado um grafo: G = ({v1,v2,...,vn},E) e k inteiro positivo

- Fase não determinística:
  - "chute" uma sequência aleatória de cores
    - Comprimento N (número de vértices), dentre k cores

Exemplo

- Fase determinística:

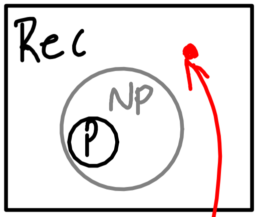
k-colorável $\leftarrow$ *true*
FOR $i = 1$ to $n - 1$ DO
    FOR $j = i + 1$ TO n DO
        IF $(v_i, v_j) \in E$ AND $C_{l_i} = C_{l_j}$ THEN
            k-colorável $\leftarrow$ *false*;
            EXIT;
RETURN k-colorável

  - Resolve-se em O(n2) com o algoritmo não determinístico apresentado

  – Similarmente, soluções para os outros problemas, podem ser “chutadas” em tempo linear e verificadas em tempo polinomial no tamanho da entrada

Não RE | RE

Teoremas



- $\mathcal{P} \subseteq \mathcal{NP}$ ✓
  - Pode-se conceber como um algoritmo não determinístico com fase inicial vazia, e fase determinística polinomialmente limitada
- $\cancel{\mathcal{NP} \subseteq \mathcal{P}}$?
  - É um dos sete problemas do milênio do Clay Mathematics Institute

Caixeiro Viajante

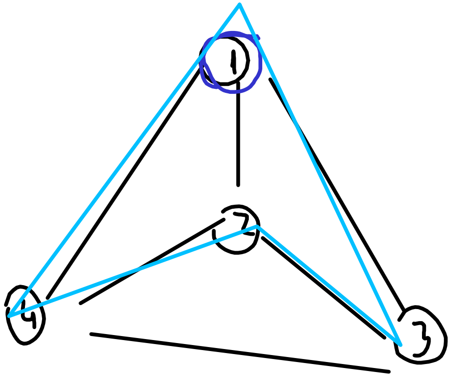
Caixeiro Viajante $\notin P$

$\in$ ou $\notin$ a NP? PQ?

Circuito Hamiltoniano $\Rightarrow \in NP$

$\notin P$

$\in$ ou $\notin$ a NP?

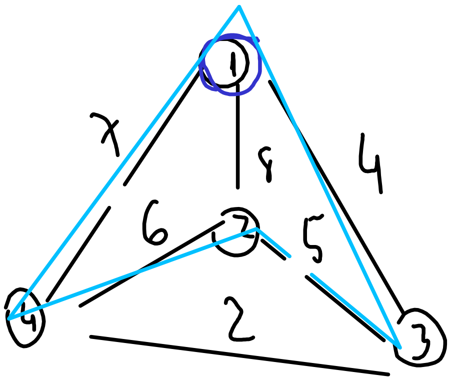


$O(n)$ — 1ª Fase: Chute

$O(n)$ — 2ª Fase: Verif

1 - 3 - 2 - 4	1 - 3 - 4 - 2	1 2 1 3
✓	✓	✗

Caixeiro Viajante $\notin$ NP



∞

O(n) — 1ª Fase: Chute

O(∞) — 2ª Fase: Verif

1

1 3 2 4	1 3 4 2	...	1 2 3 4	1 4 3 2
4 5 6 7	4 2 6 8			
$\boxed{22}$	20			