# Complexidade Computacional

# 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.

# 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.

# Problemas

- Problemas naturais, de interesse prático, não pertencem a Classe P
  - Circuito de Hamilton
  - Caixeiro Viajante
  - Satisfatibilidade
  - Coloração de Grafos
- Nenhum algoritmo de resposta polinomial foi desenvolvido até hoje

# Problemas

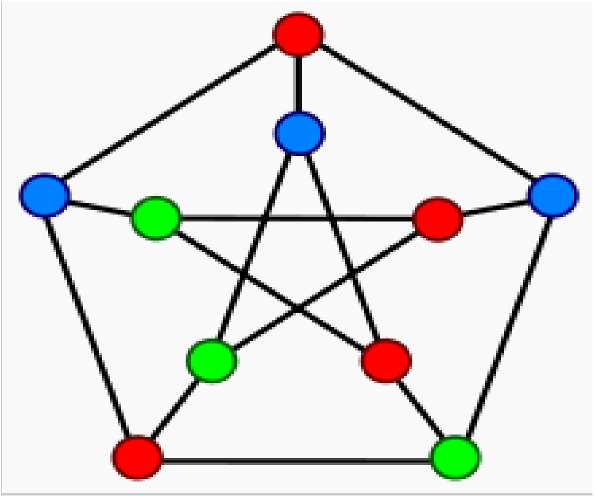
- Solução:
  - Provar que algoritmos de resposta polinomial para estes problemas não são possíveis?
- Infelizmente não!
  - Alguns desses problemas podem ser resolvidos por máquinas de Turing não-determinísticas polinomialmente limitadas

# Classe NP

- Definição:
  - 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



# Exemplo

- 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...

# 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

# Teoremas

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