Disciplinas CCM-001 / MCTA003-17
Análise de Algoritmos (e Estruturas de Dados)

Complexidade computacional

Profa. Carla Negri Lintzmayer

carla.negri@ufabc.edu.br

www.professor.ufabc.edu.br/∼carla.negri

Centro de Matemática, Computação e Cognição -- Universidade Federal do ABC

Conteúdo

Estes slides contêm uma primeira parte com foco em redução (Seção 1) e uma segunda parte com uma introdução à complexidade computacional (Seção 2).

Redução como ferramenta para resolver problemas

Redução informalmente

- Problema P: I_P é instância e S_P é solução.

Redução informalmente

- Problema P: I_P é instância e S_P é solução.
- Reduzir um problema A para um problema B significa usar um algoritmo de B para resolver A:

Redução informalmente

- Problema P: I_P é instância e S_P é solução.
- Reduzir um problema A para um problema B significa usar um algoritmo de B para resolver A:

$$I_B \longrightarrow \boxed{ALG_B} \longrightarrow S_B$$

Redução informalmente

- Problema P: I_P é instância e S_P é solução.
- Reduzir um problema A para um problema B significa usar um algoritmo de B para resolver A:

$$I_B \longrightarrow \boxed{ALG_B} \longrightarrow S_B$$

$$I_A \xrightarrow{f} I_B \longrightarrow \boxed{ALG_B} \longrightarrow S_B \xrightarrow{g} S_A$$

Redução informalmente

- Problema P: I_P é instância e S_P é solução.
- Reduzir um problema A para um problema B significa usar um algoritmo de B para resolver A:

$$I_B \longrightarrow \boxed{ALG_B} \longrightarrow S_B$$

$$I_A \xrightarrow{f} I_B \longrightarrow \boxed{ALG_B} \longrightarrow S_B \xrightarrow{g} S_A$$

$$I_A \longrightarrow \boxed{\xrightarrow{f} I_B \longrightarrow \boxed{ALG_B} \longrightarrow S_B \xrightarrow{g}} \longrightarrow S_A$$

Exemplo

PROBLEMA: SELECAO

Entrada: $\langle V, n, k\rangle$, onde $V[1..n]$ é um vetor e $k \in \{1, \ldots, n\}$.

Saída: k-ésimo menor elemento armazenado em V.

Exemplo

PROBLEMA: SELECAO

Entrada: $\langle V, n, k \rangle$, onde $V[1..n]$ é um vetor e $k \in \{1, \ldots, n\}$.
Saída: k-ésimo menor elemento armazenado em V.

PROBLEMA: ORDENACAO

Entrada: $\langle A, m \rangle$, onde $A[1..m]$ é um vetor.
Saída: vetor B com o mesmo conteúdo de A porém tal que $B[1] \leq B[2] \leq \cdots \leq B[n]$.

SELECAO reduz para ORDENACAO

Precisamos mostrar:

$$\langle V, n, k \rangle \xrightarrow{f} \langle A, m \rangle \longrightarrow \text{ Ordena } \longrightarrow A \xrightarrow{g} V[\cdot]$$

SELECAO reduz para ORDENACAO

Precisamos mostrar:

$$\langle V, n, k\rangle \xrightarrow{f} \langle A, m\rangle \longrightarrow \textsc{Ordena} \longrightarrow A \xrightarrow{g} V[\cdot]$$

```
ALG_selecao(V, n, k) {
    ALG_ordenacao(V, n);
    return V[k];
}
```

ORDENACAO reduz para SELECAO

Precisamos mostrar:

$$\langle A, m \rangle \xrightarrow{f} \langle V, n, k \rangle \longrightarrow Seleciona \longrightarrow V[\cdot] \xrightarrow{g} A$$

ORDENACAO reduz para SELECAO

Precisamos mostrar:

$$\langle A, m \rangle \xrightarrow{f} \langle V, n, k \rangle \longrightarrow Seleciona \longrightarrow V[\cdot] \xrightarrow{g} A$$

```
ALG2_ordenacao(V, n) {
    seja A[1..n] um vetor vazio;
    for (i = 1; i <= n; i++) {
        A[i] = ALG2_selecao(V, n, i);
    }
    return A;
}
```

Exemplo

PROBLEMA: TAREFAS

Entrada: $\langle T, n, s, f \rangle$, onde $T = \{t_1, \ldots, t_n\}$ é um conjunto de n tarefas onde cada $t_i \in T$ ocorre no intervalo $[s_i, f_i)$.
Saída: conjunto $S \subseteq T$ tal que as tarefas de S são mutuamente compatíveis e $|S|$ é máximo.

Exemplo

Problema: TAREFAS

Entrada: $\langle T, n, s, f \rangle$, onde $T = \{t_1, \ldots, t_n\}$ é um conjunto de n tarefas onde cada $t_i \in T$ ocorre no intervalo $[s_i, f_i)$.
Saída: conjunto $S \subseteq T$ tal que as tarefas de S são mutuamente compatíveis e $|S|$ é máximo.

Problema: CONJINDEPENDENTE

Entrada: $\langle G \rangle$, onde G é um grafo.
Saída: $I \subseteq V(G)$ tal que para todo par $u, v \in I$, $uv \notin E(G)$ e $|I|$ é máximo.

TAREFAS reduz para CONJINDEPENDENTE

Precisamos mostrar:

$$\langle T, n, s, f\rangle \xrightarrow{f} \langle G\rangle \longrightarrow ALG_{CONJINDEPENDENTE} \longrightarrow I \xrightarrow{g} S$$

TAREFAS reduz para CONJINDEPENDENTE

Precisamos mostrar:

$$\langle T, n, s, f\rangle \xrightarrow{f} \langle G\rangle \longrightarrow ALG_{CONJINDEPENDENTE} \longrightarrow I \xrightarrow{g} S$$

```
ALG_tarefas(T, n, s, f) {
    V(G) = T;
    E(G) = { {ti,tj} : s[i] < f[j] e s[j] < f[i]};
    S = ALG_conjindependente(G);
    return S;
}
```

Resumo

- Dado um problema A, podemos criar um algoritmo para A ao fazer uma redução para B, pois basta usar qualquer algoritmo para B.
- Observe que isso não impede que haja outros algoritmos para A.

Complexidade computacional

Problemas de decisão

Otimização

Dadas restrições e uma função objetivo que determina o valor de cada solução, encontrar uma solução melhor valor de função objetivo (maximização ou minimização).

Otimização

Dadas restrições e uma função objetivo que determina o valor de cada solução, encontrar uma solução melhor valor de função objetivo (maximização ou minimização).

PROBLEMA: MOCHILAOPT

Entrada: $\langle I, n, v, w, W\rangle$, onde $I = \{1, \ldots, n\}$, v_i é o valor e w_i é o peso do item $i \in I$ e W é a capacidade de peso da mochila.

Saída: subconjunto de itens $S \subseteq I$ com $\sum_{i \in S} w_i \leq W$ e $\sum_{i \in S} v_i$ máximo.

Problemas de decisão

Decisão

Dadas restrições e uma pergunta, responder `sim` ou `não`.

Decisão

Dadas restrições e uma pergunta, responder `sim` ou `não`.

PROBLEMA: MOCHILA

Entrada: $\langle I, n, v, w, W, V\rangle$, onde $I = \{1, \ldots, n\}$, v_i é o valor e w_i é o peso do item $i \in I$, W é a capacidade de peso da mochila e V é um número.

Decisão: existe um subconjunto de itens $S \subseteq I$ com $\sum_{i \in S} w_i \leq W$ e $\sum_{i \in S} v_i \geq V$?

Equivalência

- Problemas de otimização e decisão correspondentes são equivalentes no sentido de que se resolvermos um deles, então resolvemos o outro.

Equivalência

- Problemas de otimização e decisão correspondentes são equivalentes no sentido de que se resolvermos um deles, então resolvemos o outro.
    - Podem ser reduzidos entre si!

MOCHILA reduz para MOCHILAOPT

Suponha que sabemos resolver MOCHILAOPT em $\langle I, n, v, w, W \rangle$.
Conseguimos resolver MOCHILA?

MOCHILA reduz para MOCHILAOPT

Suponha que sabemos resolver MOCHILAOPT em $\langle I, n, v, w, W \rangle$. Conseguimos resolver MOCHILA?

```
ALG_mochila(I, n, v, w, W, V) {
    z = ALG_mochilaopt(I, n, v, w, W);
    if (z >= V)
        return "sim";
    else
        return "não";
}
```

MOCHILAOPT reduz para MOCHILA

Agora suponha que sabemos resolver MOCHILA em $\langle I, n, v, w, W, V \rangle$. Conseguimos resolver MOCHILAOPT?

MOCHILAOPT reduz para MOCHILA

Agora suponha que sabemos resolver MOCHILA em $\langle I, n, v, w, W, V\rangle$. Conseguimos resolver MOCHILAOPT?

```
ALG_mochilaopt(I, n, v, w, W) {
    ini = 0;
    fim = n * max{v[1], ..., v[n]};

    while (ini < fim) {
        meio = (ini + fim) / 2;
        if (ALG_mochila(I, n, v, w, W, meio) == "sim")
            ini = meio + 1;
        else
            fim = meio - 1;
    }

    if (ALG_mochila(I, n, v, w, W, ini) == "sim")
        return ini;
    else
        return ini - 1;
}
```

Outros problemas de decisão

PROBLEMA: ALINHAMENTO

Entrada: $\langle X, m, Y, n, \alpha, p\rangle$, onde $X[1..m]$ e $Y = [1..n]$ são sequências, α é função de pontuação e p é um número.
Decisão: existe um alinhamento de X e Y com pontuação $\geq p$?

Problema: ARVOREGERADORA

Entrada: $\langle G, w, W \rangle$, onde G é um grafo, $w\colon E(G) \to \mathbb{R}$ e W é um número.

Decisão: existe conjunto $T \subseteq E(G)$ tal que $G[T]$ é árvore geradora e $\sum_{e \in T} w(e) \leq W$?

Outros problemas de decisão

PROBLEMA: BARRA

Entrada: $\langle n, p, k \rangle$, onde n é um inteiro tamanho da barra, vender um pedaço de tamanho i dá lucro p_i e k é um número.
Decisão: existe forma de cortar a barra tal que a soma dos lucros de vendas é $\geq k$?

Outros problemas de decisão

PROBLEMA: CAMINHOHAMILT

Entrada: $\langle D \rangle$, onde D é um digrafo.

Decisão: existe caminho hamiltoniano em D?

Outros problemas de decisão

PROBLEMA: CAMINHOHAMILT

Entrada: $\langle D \rangle$, onde D é um digrafo.

Decisão: existe caminho hamiltoniano em D?

PROBLEMA: CICLOHAMILT

Entrada: $\langle D \rangle$, onde D é um digrafo.

Decisão: existe ciclo hamiltoniano em D?

Outros problemas de decisão

Problema: CAMINHOHAMILT

Entrada: $\langle D \rangle$, onde D é um digrafo.
Decisão: existe caminho hamiltoniano em D?

Problema: CICLOHAMILT

Entrada: $\langle D \rangle$, onde D é um digrafo.
Decisão: existe ciclo hamiltoniano em D?

Problema: TSP

Entrada: $\langle D, w, k \rangle$, onde D é um digrafo, $w \colon E(D) \to \mathbb{R}$ e $k \in \mathbb{R}$.
Decisão: existe ciclo hamiltoniano em D com peso $\leq k$?

Redução

Redução formalmente

Redução polinomial

Sejam A e B dois problemas de decisão.

O problema A é *redutível* para B se existe um algoritmo eficiente f tal que $f(I_A) = I_B$ e

$$I_A \text{ é } \texttt{sim} \text{ se e somente se } I_B \text{ é } \texttt{sim}.$$

CAMINHOHAMILT reduz polinomialmente para CICLOHAMILT

Seja $\langle D \rangle$ uma entrada de CAMINHOHAMILT.

CAMINHOHAMILT reduz polinomialmente para CICLOHAMILT

Seja $\langle D \rangle$ uma entrada de CAMINHOHAMILT.

Construa $\langle D' \rangle$ onde $V(D') = V(D) \cup \{x\}$ e
$E(D') = E(D) \cup \{xu, ux \colon u \in V(D)\}$ (isso leva tempo polinomial).

CAMINHOHAMILT reduz polinomialmente para CICLOHAMILT

Seja $\langle D \rangle$ uma entrada de CAMINHOHAMILT.

Construa $\langle D' \rangle$ onde $V(D') = V(D) \cup \{x\}$ e
$E(D') = E(D) \cup \{xu, ux \colon u \in V(D)\}$ (isso leva tempo polinomial).

Resta mostrar que $\langle D \rangle$ é `sim` para CAMINHOHAMILT se e somente se $\langle D' \rangle$ é `sim` para CICLOHAMILT.

CAMINHOHAMILT reduz polinomialmente para CICLOHAMILT

Seja $\langle D \rangle$ uma entrada de CAMINHOHAMILT.

Construa $\langle D' \rangle$ onde $V(D') = V(D) \cup \{x\}$ e
$E(D') = E(D) \cup \{xu, ux \colon u \in V(D)\}$ (isso leva tempo polinomial).

Resta mostrar que $\langle D \rangle$ é `sim` para CAMINHOHAMILT se e somente se $\langle D' \rangle$ é `sim` para CICLOHAMILT.

Suponha primeiro que $\langle D \rangle$ é `sim` para CAMINHOHAMILT.
Então existe caminho $C = (a, b, \ldots, d)$ hamiltoniano em D.
Note que $C' = (a, b, \ldots, d, x, a)$ é ciclo hamiltoniano em D'.
Então $\langle D' \rangle$ é `sim` para CICLOHAMILT.

CAMINHOHAMILT reduz polinomialmente para CICLOHAMILT

Seja $\langle D \rangle$ uma entrada de CAMINHOHAMILT.

Construa $\langle D' \rangle$ onde $V(D') = V(D) \cup \{x\}$ e
$E(D') = E(D) \cup \{xu, ux \colon u \in V(D)\}$ (isso leva tempo polinomial).

Resta mostrar que $\langle D \rangle$ é `sim` para CAMINHOHAMILT se e somente se $\langle D' \rangle$ é `sim` para CICLOHAMILT.

Suponha primeiro que $\langle D \rangle$ é `sim` para CAMINHOHAMILT.
Então existe caminho $C = (a, b, \ldots, d)$ hamiltoniano em D.
Note que $C' = (a, b, \ldots, d, x, a)$ é ciclo hamiltoniano em D'.
Então $\langle D' \rangle$ é `sim` para CICLOHAMILT.

Suponha agora que $\langle D' \rangle$ é `sim` para CICLOHAMILT.

Então existe ciclo $D = (a, \ldots, b, x, d, \ldots, a)$ hamiltoniano em D'.

Note que $C' = (d, \ldots, a, \ldots, b)$ é caminho hamiltoniano em D.

Então $\langle D \rangle$ é `sim` para CAMINHOHAMILT.

BARRA reduz polinomialmente para MOCHILA

Seja $\langle n, p, k \rangle$ uma entrada de BARRA.

BARRA reduz polinomialmente para MOCHILA

Seja $\langle n, p, k\rangle$ uma entrada de BARRA.

Construa $\langle I, m, v, w, W, V\rangle$ da seguinte forma:

- crie $\lfloor n/i\rfloor$ itens de peso $w = i$ e valor $v = p_i$, correspondentes a um pedaço de tamanho i da barra, $1 \leq i \leq n$;
- $m = \sum_{i=1}^{n} \lfloor n/i\rfloor$ e $I = \{1, \ldots, m\}$;
- $W = n$;
- $V = k$.

Note que isso é feito em tempo polinomial.

BARRA reduz polinomialmente para MOCHILA

Seja $\langle n, p, k\rangle$ uma entrada de BARRA.

Construa $\langle I, m, v, w, W, V\rangle$ da seguinte forma:

- crie $\lfloor n/i \rfloor$ itens de peso $w = i$ e valor $v = p_i$, correspondentes a um pedaço de tamanho i da barra, $1 \leq i \leq n$;
- $m = \sum_{i=1}^{n} \lfloor n/i \rfloor$ e $I = \{1, \dots, m\}$;
- $W = n$;
- $V = k$.

Note que isso é feito em tempo polinomial.

Resta mostrar que $\langle n, p, k\rangle$ é `sim` para BARRA se e somente se $\langle I, m, v, w, W, V\rangle$ é `sim` para MOCHILA.

BARRA reduz polinomialmente para MOCHILA

Suponha que $\langle n, p, k \rangle$ é `sim` para BARRA.

Então existem pedaços $(c_1, c_2, \ldots, c_x)$ tais que $\sum_{i=1}^{x} c_i = n$ e $\sum_{i=1}^{x} p_{c_i} \geq k$.
Para cada pedaço c_i, coloque o item j correspondente ao mesmo em um conjunto S.
Note que $\sum_{j \in S} w_j = \sum_{i=1}^{x} c_i = n = W$ e $\sum_{j \in S} v_j = \sum_{i=1}^{x} p_{c_i} \geq k = V$.

Então $\langle I, m, v, w, W, V \rangle$ é `sim` para MOCHILA.

BARRA reduz polinomialmente para MOCHILA

Suponha agora que $\langle I, m, v, w, W, V\rangle$ é `sim` para MOCHILA.

Então existe conjunto $S \subseteq I$ de itens tais que $\sum_{j \in S} w_j \leq W$ e $\sum_{j \in S} v_j \geq V$.
Para cada item $j \in S$, corte a barra em um tamanho i correspondente ao mesmo.
Sejam $(c_1, c_2, \ldots, c_{|S|})$ os pedaços cortados da barra.
Note que $\sum_{i=1}^{|S|} c_i = \sum_{j \in S} w_j \leq W = n$ e $\sum_{i=1}^{|S|} p_{c_i} = \sum_{j \in S} v_j \geq V = k$.

Então $\langle n, p, k\rangle$ é `sim` para BARRA.

Outro problema

PROBLEMA: SUBSETSUM

Entrada: $\langle A, n, B \rangle$, onde $A = \{s_1, \ldots, s_n\}$ é um conjunto de n inteiros e B é um inteiro.

Decisão: existe $A' \subseteq A$ tal que $\sum_{s \in A'} s = B$?

SUBSETSUM reduz polinomialmente para MOCHILA

Seja $\langle A, n, B \rangle$ uma entrada de MOCHILA.

SUBSETSUM reduz polinomialmente para MOCHILA

Seja $\langle A, n, B \rangle$ uma entrada de MOCHILA.

Construa $\langle I, m, v, w, W, V \rangle$ da seguinte forma:

- crie um item de peso $w = s_i$ e valor $v = s_i$ para cada $s_i \in A$;
- $m = n$ e $I = \{1, \ldots, n\}$;
- $W = B$;
- $V = B$.

Note que isso é feito em tempo polinomial.

SUBSETSUM reduz polinomialmente para MOCHILA

Seja $\langle A, n, B \rangle$ uma entrada de MOCHILA.

Construa $\langle I, m, v, w, W, V \rangle$ da seguinte forma:

- crie um item de peso $w = s_i$ e valor $v = s_i$ para cada $s_i \in A$;
- $m = n$ e $I = \{1, \ldots, n\}$;
- $W = B$;
- $V = B$.

Note que isso é feito em tempo polinomial.

Resta mostrar que $\langle A, n, B \rangle$ é `sim` para SUBSETSUM se e somente se $\langle I, m, v, w, W, V \rangle$ é `sim` para MOCHILA.

SUBSETSUM reduz polinomialmente para MOCHILA

Suponha que $\langle A, n, B \rangle$ é `sim` para SUBSETSUM.

Então existe $A' \subseteq A$ com $\sum_{s \in A'} s = B$.
Para cada $s \in A'$, coloque o item j correspondente em um conjunto S.
Note que $\sum_{j \in S} w_j = \sum_{s \in A'} s = B = W$ e $\sum_{j \in S} v_j = \sum_{s \in A'} s = B = V$.

Então $\langle I, m, v, w, W, V \rangle$ é `sim` para MOCHILA.

SUBSETSUM reduz polinomialmente para MOCHILA

Agora suponha que $\langle I, m, v, w, W, V\rangle$ é `sim` para MOCHILA.

Então existe $S \subseteq I$ com $\sum_{j \in S} w_j \leq W$ e $\sum_{j \in S} v_j \geq V$.
Para cada $j \in S$, coloque o valor s_i correspondente em um conjunto A'.
Note que $\sum_{s \in A'} s = \sum_{j \in S} w_j \leq W = B$ e $\sum_{s \in A'} s = \sum_{j \in S} v_j \geq V = B$.
Mas então só pode ser que $\sum_{s \in A'} s = B$.

Então $\langle A, n, B\rangle$ é `sim` para SUBSETSUM.

Mas qual a importância da redução?

Além de ser uma técnica de projeto de algoritmos, ela é muito utilizada para dar informações sobre a dificuldade de problemas.

Suponha que o problema A reduz em tempo polinomial para o problema B:

$$I_A \longrightarrow \boxed{\xrightarrow{f} I_B \longrightarrow \boxed{ALG_B} \longrightarrow S_B \xrightarrow{g}} \longrightarrow S_A$$

Mas qual a importância da redução?

Além de ser uma técnica de projeto de algoritmos, ela é muito utilizada para dar informações sobre a dificuldade de problemas.

Suponha que o problema A reduz em tempo polinomial para o problema B:

$$I_A \longrightarrow \boxed{\xrightarrow{f} I_B \longrightarrow \boxed{ALG_B} \longrightarrow S_B \xrightarrow{g}} \longrightarrow S_A$$

Que conclusão podemos tirar se:

1. existir algoritmo eficiente para A?
2. existir algoritmo eficiente para B?
3. não existir algoritmo eficiente para A?
4. não existir algoritmo eficiente para B?

Mas qual a importância da redução?

Além de ser uma técnica de projeto de algoritmos, ela é muito utilizada para dar informações sobre a dificuldade de problemas.

Suponha que o problema A reduz em tempo polinomial para o problema B:

$$I_A \longrightarrow \boxed{\xrightarrow{f} I_B \longrightarrow \boxed{ALG_B} \longrightarrow S_B \xrightarrow{g}} \longrightarrow S_A$$

Que conclusão podemos tirar se:

1. existir algoritmo eficiente para A?
2. existir algoritmo eficiente para B?
3. não existir algoritmo eficiente para A?
4. não existir algoritmo eficiente para B?

Ou seja, B é tão difícil quanto A.
Ou então, A não é mais difícil do que B.

Mas qual a importância da redução?

Com exemplos:

- BARRA reduz polinomialmente para MOCHILA.
- SUBSETSUM reduz polinomialmente para MOCHILA.

Sabemos que:

- existe algoritmo eficiente para BARRA;
- (ainda) não existe algoritmo eficiente para SUBSETSUM.

Qual a conclusão sobre MOCHILA?

Classes de complexidade

Classe P

Classe P

Conjunto de problemas de decisão que podem ser resolvidos por um algoritmo eficiente.

BARRA, ALINHAMENTO, ARVOREGERADORA, CAMINHOCURTO são problemas em P.

Nem todos os problemas do mundo estão em P

Não se sabe!

Até agora, não encontramos algoritmos de tempo polinomial que resolvem MOCHILA, TSP, CICLOHAMILT, CAMINHOHAMILT, SUBSETSUM, ...

Mas eles são facilmente verificáveis

Para convencer alguém que uma instância é `sim`, basta apresentar um *certificado positivo*.

- $\langle\{2,4,5,7,9,10\},5,9\rangle$ é `sim` para SUBSETSUM?
- $\langle\{2,4,5,7,9,10\},5,25\rangle$ é `sim` para SUBSETSUM?
- $\langle\{1,2,3,4,5\},5,v=(3,4,6,21,1),w=(3,4,6,2,7),10,15\rangle$ é `sim` para MOCHILA?
- $\langle\{1,2,3,4,5\},5,v=(3,4,6,21,1),w=(3,4,6,2,7),10,48\rangle$ é `sim` para MOCHILA?

Algoritmo verificador

Algoritmo verificador

Seja T um problema qualquer.

Um algoritmo A é *verificador* se:

- para toda I_T que é `sim`, existe um conjunto de dados D tal que $A(I_T, D)$ retorna `sim` em tempo polinomial;
- para toda I_T que é `não`, qualquer conjunto de dados D faz $A(I_T, D)$ retornar `não` em tempo polinomial.

O conjunto D acima é um *certificado positivo*.

Classe NP

Conjunto de problemas de decisão para os quais existe um algoritmo verificador que aceita um certificado positivo em tempo polinomial.

É o caso de MOCHILA, TSP, CICLOHAMILT, CAMINHOHAMILT, SUBSETSUM, STEINER, ...

P vs. NP

Note que $\mathsf{P} \subseteq \mathsf{NP}$.

P vs. NP

Note que $\mathsf{P} \subseteq \mathsf{NP}$.

Será que $\mathsf{NP} \subseteq \mathsf{P}$?

P vs. NP

Note que P $\subseteq$ NP.

Será que NP $\subseteq$ P?

P é igual a NP?

P vs. NP

Note que P $\subseteq$ NP.

Será que NP $\subseteq$ P?

P é igual a NP?

Um dos *problemas do milênio*: prêmio de 1 milhão de dólares para quem resolver.

Problemas NP-completos

Discussão

- O que significa dizer que um problema A está em NP?

Problemas NP-completos

Problema NP-completo

Um problema Q é NP-completo se $Q \in \text{NP}$ e todo outro problema de NP se reduz polinomialmente a Q.

O que isso significa?

Problema NP-completo

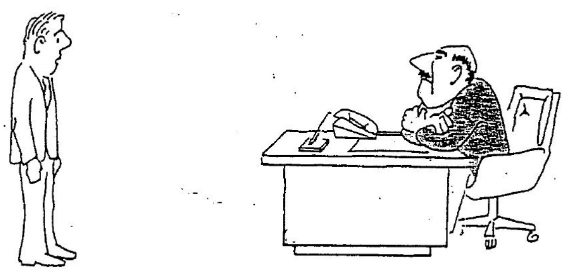
Um problema Q é NP-completo se $Q \in$ NP e todo outro problema de NP se reduz polinomialmente a Q.

O que isso significa?

Se X é NP-completo, podemos dizer que P $=$ NP se e somente se X pertence a P.

Resolva esse problema

Ele está em NP!



``I can't find an efficient algorithm, I guess I'm just too dumb.''

Problemas NP-completos

Ele é NP-completo!

``I can't find an efficient algorithm, but neither can all these famous people.''

Problemas NP-difíceis

Problema NP-difícil

Um problema Q é NP-difícil se todo outro problema de NP se reduz polinomialmente a Q.

Problemas NP-difíceis

Problema NP-difícil

Um problema Q é NP-difícil se todo outro problema de NP se reduz polinomialmente a Q.

Problema NP-completo

Um problema Q é NP-completo se $Q \in$ NP e Q é NP-difícil.

E como mostrar que um problema é NP-completo?

Problemas NP-completos

E como mostrar que um problema é NP-completo?

Primeiro mostramos que o problema está em NP.

Problemas NP-completos

E como mostrar que um problema é NP-completo?

Primeiro mostramos que o problema está em NP.

Depois enumeramos todos os problemas de NP e fazemos uma redução polinomial de cada um deles para o nosso problema (mostramos que ele é NP-difícil).

Problemas NP-completos

E como mostrar que um problema é NP-completo?

Primeiro mostramos que o problema está em NP.

Depois enumeramos todos os problemas de NP e fazemos uma redução polinomial de cada um deles para o nosso problema (mostramos que ele é NP-difícil).

Mas a operação de redução pode ser composta!

Problemas NP-completos

E como mostrar que um problema é NP-completo?

E como mostrar que um problema é NP-completo?

Primeiro mostramos que o problema está em NP.

Problemas NP-completos

E como mostrar que um problema é NP-completo?

Primeiro mostramos que o problema está em NP.

Depois encontramos um problema que já é NP-completo e o reduzimos polinomialmente para o nosso.

Problemas NP-completos

- Vimos que MOCHILA está em NP.
- Vimos que SUBSETSUM reduz polinomialmente para MOCHILA.
- Sabemos que SUBSETSUM É NP-completo.
- Portanto, MOCHILA é NP-completo!

Problemas NP-completos

- Vimos que MOCHILA está em NP.
- Vimos que SUBSETSUM reduz polinomialmente para MOCHILA.
- Sabemos (??) que SUBSETSUM É NP-completo.
- Portanto, MOCHILA é NP-completo!

Mas e o primeiro problema NP-completo?

Mas e o primeiro problema NP-completo?

Problema: 3-SAT

Entrada: $\langle \phi \rangle$, onde ϕ é uma fórmula 3-CNF contendo literais de variáveis booleanas $x_1, \ldots, x_n$.

Decisão: Existe uma atribuição de valores a $x_1, \ldots, x_n$ tal que ϕ é satisfatível?

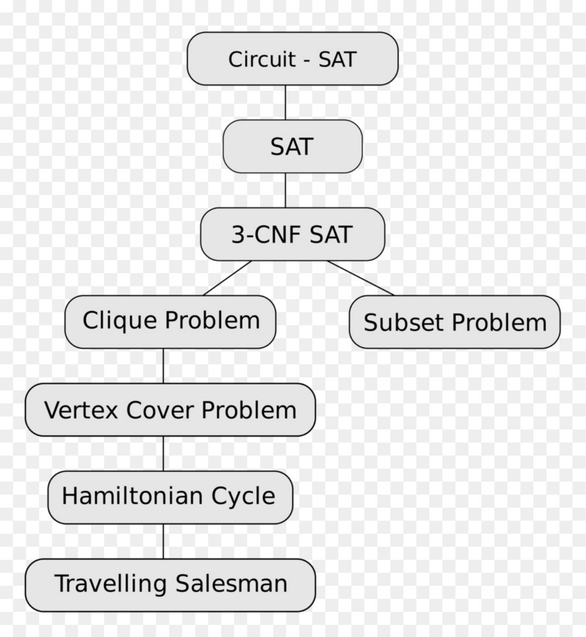
Mas e o primeiro problema NP-completo?

PROBLEMA: 3-SAT

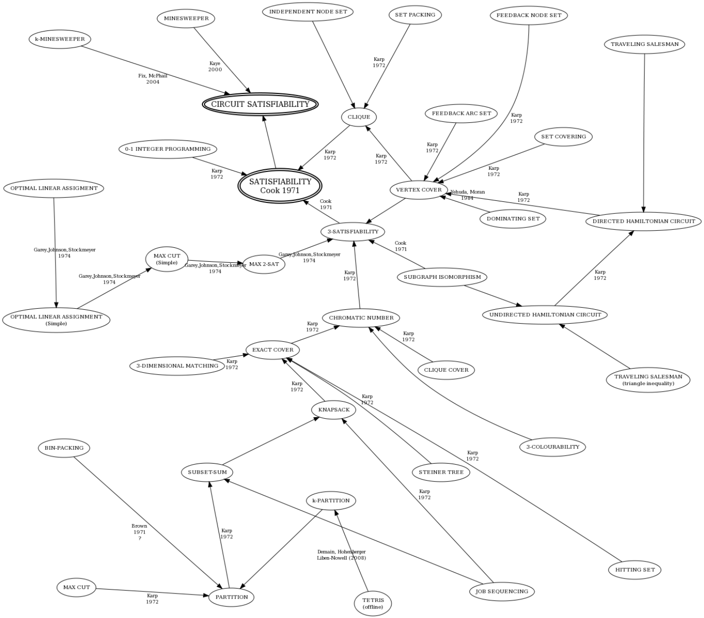
Entrada: $\langle \phi \rangle$, onde ϕ é uma fórmula 3-CNF contendo literais de variáveis booleanas $x_1, \ldots, x_n$.
Decisão: Existe uma atribuição de valores a $x_1, \ldots, x_n$ tal que ϕ é satisfatível?

Teorema de Cook-Levin: 3-SAT é NP-completo.

Problemas NP-completos



Problemas NP-completos



Outro problema NP-completo

Problema: CLIQUE

Entrada: $\langle G, k \rangle$, onde G é um grafo e k é um inteiro positivo.
Decisão: Existe conjunto $S \subseteq V(G)$ tal que para todo par $u, v \in S$ existe aresta $uv \in E(G)$ e $|S| \geq k$?

Outro problema NP-completo

PROBLEMA: CLIQUE

Entrada: $\langle G, k \rangle$, onde G é um grafo e k é um inteiro positivo.
Decisão: Existe conjunto $S \subseteq V(G)$ tal que para todo par $u, v \in S$ existe aresta $uv \in E(G)$ e $|S| \geq k$?

Teorema

CLIQUE é NP-completo.

Demonstração

Primeiro mostramos que CLIQUE está em NP e depois faremos uma redução polinomial de 3-SAT para CLIQUE.

Teorema

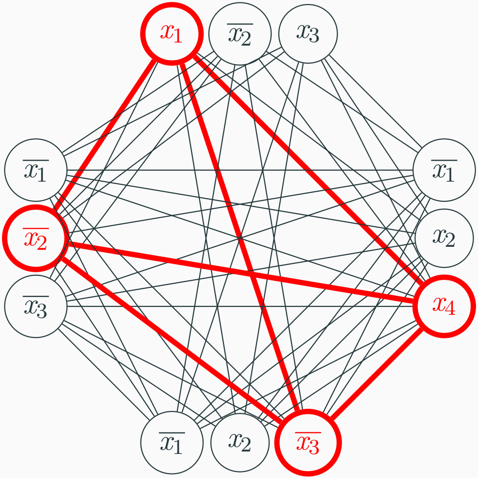
CLIQUE é NP-completo.

Demonstração (continuação)

CLIQUE está em NP pois, dados $\langle G, k \rangle$ e um conjunto S qualquer de vértices, é fácil escrever um algoritmo eficiente que verifica se S é uma clique de tamanho pelo menos k: basta verificar se todos os pares de vértices em S formam arestas e contar a quantidade de vértices em S.

Outro problema NP-completo

Dado $\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_4) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$, construa:



Outro problema NP-completo

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Agora considere uma fórmula 3-CNF ϕ com m cláusulas sobre literais das variáveis $v_1, \ldots, v_n$.

Outro problema NP-completo

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Agora considere uma fórmula 3-CNF ϕ com m cláusulas sobre literais das variáveis $v_1, \ldots, v_n$.

Vamos construir um grafo G com $3m$ vértices: para cada cláusula, temos 3 vértices representando seus literais.

Outro problema NP-completo

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Agora considere uma fórmula 3-CNF ϕ com m cláusulas sobre literais das variáveis $v_1, \ldots, v_n$.

Vamos construir um grafo G com $3m$ vértices: para cada cláusula, temos 3 vértices representando seus literais.

Haverá aresta entre dois vértices v e w se e somente se os literais representados por eles estiverem em cláusulas diferentes, v corresponde a um literal x e w não corresponde ao literal $\bar{x}$.

Outro problema NP-completo

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Agora considere uma fórmula 3-CNF ϕ com m cláusulas sobre literais das variáveis $v_1, \ldots, v_n$.

Vamos construir um grafo G com $3m$ vértices: para cada cláusula, temos 3 vértices representando seus literais.

Haverá aresta entre dois vértices v e w se e somente se os literais representados por eles estiverem em cláusulas diferentes, v corresponde a um literal x e w não corresponde ao literal $\bar{x}$.

Por fim, tome $k = m$.

Outro problema NP-completo

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Agora considere uma fórmula 3-CNF ϕ com m cláusulas sobre literais das variáveis $v_1, \ldots, v_n$.
Vamos construir um grafo G com $3m$ vértices: para cada cláusula, temos 3 vértices representando seus literais.
Haverá aresta entre dois vértices v e w se e somente se os literais representados por eles estiverem em cláusulas diferentes, v corresponde a um literal x e w não corresponde ao literal $\bar{x}$.
Por fim, tome $k = m$.
Temos então uma instância $\langle G, k \rangle$ para CLIQUE gerada em tempo polinomial. Resta verificar se ϕ é satisfatível se e somente se G contém uma clique com $\geq k = m$ vértices.

Outro problema NP-completo

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Suponha que ϕ é satisfatível.

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Suponha que ϕ é satisfatível.

Então em cada cláusula, pelo menos um literal tem valor 1.

Outro problema NP-completo

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Suponha que ϕ é satisfatível.

Então em cada cláusula, pelo menos um literal tem valor 1.

Como um literal e sua negação não podem ter valor 1, existe uma aresta entre quaisquer dois vértices que representam esses literais em G.

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Suponha que ϕ é satisfatível.

Então em cada cláusula, pelo menos um literal tem valor 1.

Como um literal e sua negação não podem ter valor 1, existe uma aresta entre quaisquer dois vértices que representam esses literais em G.

Então eles formam uma clique em G com pelo menos $m = k$ vértices.

Outro problema NP-completo

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Agora suponha que existe conjunto $S \subseteq V(G)$ que é clique e $|S| \geq k$.

Outro problema NP-completo

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Agora suponha que existe conjunto $S \subseteq V(G)$ que é clique e $|S| \geq k$.

Então existe uma aresta entre quaisquer dois vértices de S.

Outro problema NP-completo

Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Agora suponha que existe conjunto $S \subseteq V(G)$ que é clique e $|S| \geq k$.

Então existe uma aresta entre quaisquer dois vértices de S.

Então esses dois não representam literais que são a negação um do outro e eles estão em cláusulas diferentes.

Outro problema NP-completo

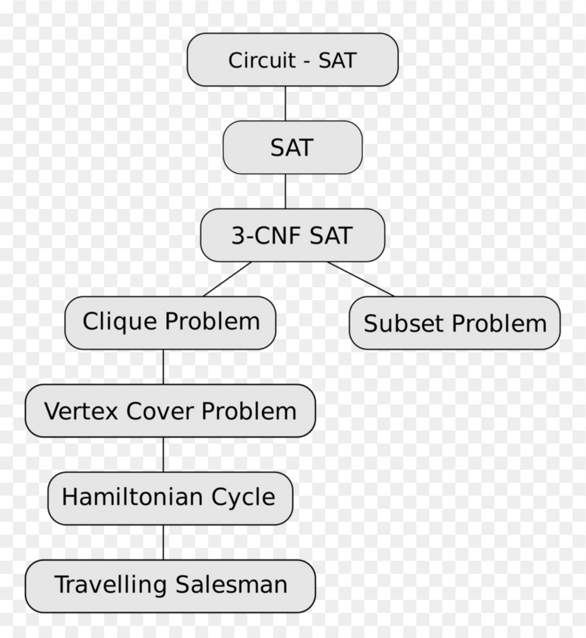
Teorema

CLIQUE é NP-completo.

Demonstração (continuação)

Agora suponha que existe conjunto $S \subseteq V(G)$ que é clique e $|S| \geq k$.
Então existe uma aresta entre quaisquer dois vértices de S.
Então esses dois não representam literais que são a negação um do outro e eles estão em cláusulas diferentes.
Então dar valor 1 aos literais representados pelos vértices de S satisfaz ϕ.

Classe NP-completo



Outro problema NP-completo

PROBLEMA: VERTEXCOVER

Entrada: $\langle G, k \rangle$, onde G é um grafo e k é um inteiro.
Decisão: Existe conjunto $S \subseteq V(G)$ (uma cobertura) tal que toda aresta $uv \in E(G)$ tem $u \in S$ ou $v \in S$ e $|S| \leq k$?

Outro problema NP-completo

PROBLEMA: VERTEXCOVER

Entrada: $\langle G, k \rangle$, onde G é um grafo e k é um inteiro.
Decisão: Existe conjunto $S \subseteq V(G)$ (uma cobertura) tal que toda aresta $uv \in E(G)$ tem $u \in S$ ou $v \in S$ e $|S| \leq k$?

Teorema

VERTEXCOVER é NP-completo.

Demonstração

Primeiro mostramos que VERTEXCOVER está em NP e depois faremos uma redução polinomial de CLIQUE para VERTEXCOVER.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

VERTEXCOVER está em NP pois, dados $\langle G, k\rangle$ e um conjunto S qualquer de vértices, é fácil escrever um algoritmo eficiente que verifica se S é uma cobertura por vértices das arestas de G que tem tamanho no máximo k: basta verificar se todas as arestas de G tem algum extremo em S e contar a quantidade de vértices em S.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Agora considere um grafo G e um valor k inteiro, instância para CLIQUE.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Agora considere um grafo G e um valor k inteiro, instância para CLIQUE.

O grafo complemento de G é o grafo $\overline{G}$, em que $V(\overline{G}) = V(G)$ e $E(\overline{G}) = \{uv\colon uv \notin E(G)\}$.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Agora considere um grafo G e um valor k inteiro, instância para CLIQUE.

O grafo complemento de G é o grafo $\overline{G}$, em que $V(\overline{G}) = V(G)$ e $E(\overline{G}) = \{uv\colon uv \notin E(G)\}$.

Tomando $t = |V(G)| - k$, temos então uma instância $\langle \overline{G}, t \rangle$ para VERTEXCOVER construída em tempo polinomial. Resta verificar se G contém uma clique de tamanho pelo menos k se e somente se $\overline{G}$ contém uma cobertura por vértices de tamanho no máximo $t = |V(G)| - k$.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Suponha que G contém uma clique S de tamanho pelo menos k.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Suponha que G contém uma clique S de tamanho pelo menos k. Isso significa que para todo par $u, v \in S$ temos $uv \in E(G)$, o que implica em $uv \notin E(\overline{G})$.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Suponha que G contém uma clique S de tamanho pelo menos k.
Isso significa que para todo par $u, v \in S$ temos $uv \in E(G)$, o que implica em $uv \notin E(\overline{G})$.
Então para toda aresta $xy \in E(\overline{G})$, devemos ter que $x \notin S$ ou $y \notin S$.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Suponha que G contém uma clique S de tamanho pelo menos k.
Isso significa que para todo par $u, v \in S$ temos $uv \in E(G)$, o que implica em $uv \notin E(\overline{G})$.
Então para toda aresta $xy \in E(\overline{G})$, devemos ter que $x \notin S$ ou $y \notin S$.
Logo, $V(G) \setminus S$ é uma cobertura por vértices de $\overline{G}$.

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Suponha que G contém uma clique S de tamanho pelo menos k.
Isso significa que para todo par $u, v \in S$ temos $uv \in E(G)$, o que implica em $uv \notin E(\overline{G})$.
Então para toda aresta $xy \in E(\overline{G})$, devemos ter que $x \notin S$ ou $y \notin S$.
Logo, $V(G) \setminus S$ é uma cobertura por vértices de $\overline{G}$.
Como $|S| \geq k$, temos $|V(G) \setminus S| = |V(G)| - |S| \leq |V(G)| - k = t$.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Agora suponha que $\overline{G}$ contém uma cobertura por vértices S de tamanho no máximo t.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Agora suponha que $\overline{G}$ contém uma cobertura por vértices S de tamanho no máximo t.

Isso significa que para toda aresta $uv \in E(\overline{G})$, temos $u \in S$ ou $v \in S$.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Agora suponha que $\overline{G}$ contém uma cobertura por vértices S de tamanho no máximo t.

Isso significa que para toda aresta $uv \in E(\overline{G})$, temos $u \in S$ ou $v \in S$.

De forma equivalente, para qualquer par de vértices x, y tais que $x \notin S$ e $y \notin S$, devemos ter $xy \notin E(\overline{G})$, o que implica em $xy \in E(G)$.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Agora suponha que $\overline{G}$ contém uma cobertura por vértices S de tamanho no máximo t.

Isso significa que para toda aresta $uv \in E(\overline{G})$, temos $u \in S$ ou $v \in S$.

De forma equivalente, para qualquer par de vértices x, y tais que $x \notin S$ e $y \notin S$, devemos ter $xy \notin E(\overline{G})$, o que implica em $xy \in E(G)$.

Logo, $V(G) \setminus S$ é uma clique em G.

Outro problema NP-completo

Teorema

VERTEXCOVER é NP-completo.

Demonstração (continuação)

Agora suponha que $\overline{G}$ contém uma cobertura por vértices S de tamanho no máximo t.

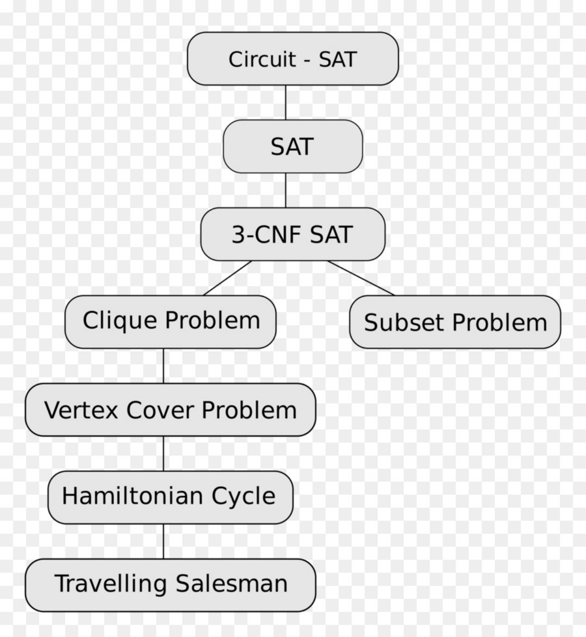
Isso significa que para toda aresta $uv \in E(\overline{G})$, temos $u \in S$ ou $v \in S$.

De forma equivalente, para qualquer par de vértices x, y tais que $x \notin S$ e $y \notin S$, devemos ter $xy \notin E(\overline{G})$, o que implica em $xy \in E(G)$.

Logo, $V(G) \setminus S$ é uma clique em G.

Como $S \leq t = |V(G)| - k$, temos $|V(G) \setminus S| = |V(G)| - |S| \geq |V(G)| - (|V(G)| - k) = k$.

Classe NP-completo



Outro problema NP-completo

Problema: CICLOHAMILT

Entrada: $\langle D \rangle$, onde D é um digrafo.

Decisão: existe ciclo hamiltoniano em D?

Outro problema NP-completo

PROBLEMA: CICLOHAMILT

Entrada: $\langle D \rangle$, onde D é um digrafo.

Decisão: existe ciclo hamiltoniano em D?

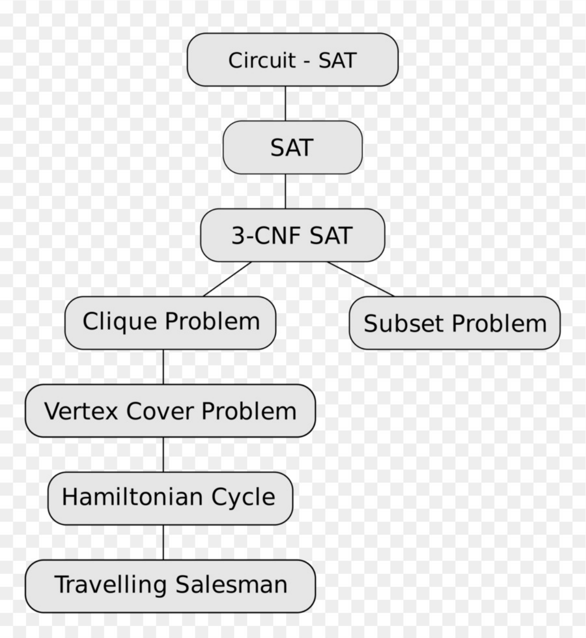
Teorema

CICLOHAMILT é NP-completo.

Ideia da demonstração

Primeiro mostramos que CICLOHAMILT está em NP e depois fazemos uma redução polinomial de VERTEXCOVER para CICLOHAMILT.

Classe NP-completo



Outro problema NP-completo

Problema: TSP

Entrada: $\langle D, w, k \rangle$, onde D é um digrafo, $w\colon E(D) \to \mathbb{R}$ e $k \in \mathbb{R}$.

Decisão: existe ciclo hamiltoniano em D com peso $\leq k$?

Outro problema NP-completo

PROBLEMA: TSP

Entrada: $\langle D, w, k\rangle$, onde D é um digrafo, w: $E(D) \rightarrow \mathbb{R}$ e $k \in \mathbb{R}$.

Decisão: existe ciclo hamiltoniano em D com peso $\leq k$?

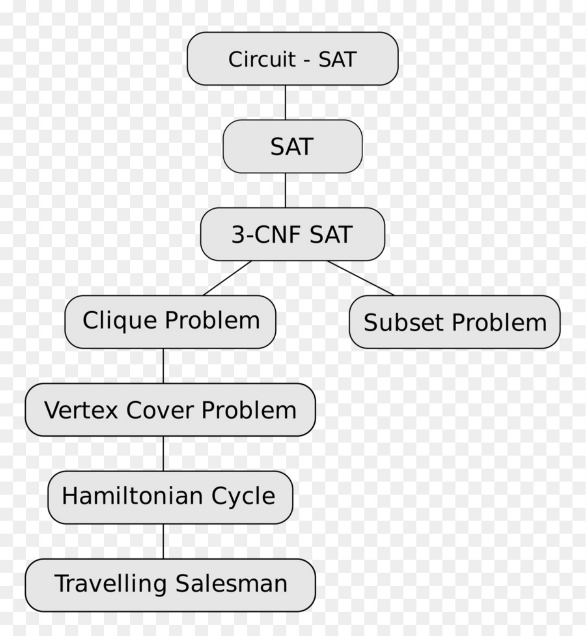
Teorema

TSP é NP-completo.

Ideia da demonstração

Primeiro mostramos que TSP está em NP e depois fazemos uma redução polinomial de CICLOHAMILT para TSP.

Classe NP-completo



Outro problema NP-completo

Problema: CAMINHOHAMILT

Entrada: $\langle D \rangle$, onde D é um digrafo.

Decisão: existe caminho hamiltoniano em D?

Outro problema NP-completo

PROBLEMA: CAMINHOHAMILT

Entrada: $\langle D \rangle$, onde D é um digrafo.

Decisão: existe caminho hamiltoniano em D?

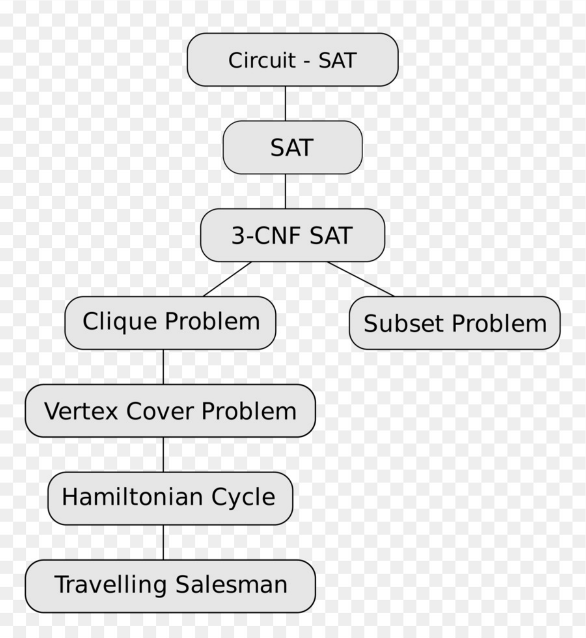
Teorema

CAMINHOHAMILT é NP-completo.

Ideia da demonstração

Primeiro mostramos que CAMINHOHAMILT está em NP e depois fazemos uma redução polinomial de CICLOHAMILT para CAMINHOHAMILT.

Classe NP-completo



Outro problema NP-completo

PROBLEMA: CAMINHOLONGO

Entrada: $\langle D, k \rangle$, onde D é um digrafo e k é um inteiro positivo.

Decisão: existe caminho em D com número de arcos $\geq k$?

Outro problema NP-completo

PROBLEMA: CAMINHOLONGO

Entrada: $\langle D, k \rangle$, onde D é um digrafo e k é um inteiro positivo.
Decisão: existe caminho em D com número de arcos $\geq k$?

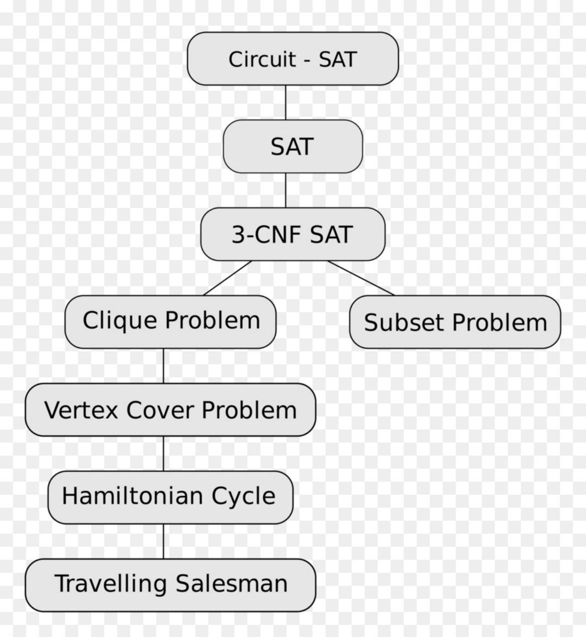
Teorema

CAMINHOLONGO é NP-completo.

Ideia da demonstração

Primeiro mostramos que CAMINHOLONGO está em NP e depois fazemos uma redução polinomial de CAMINHOHAMILT para CAMINHOLONGO.

Classe NP-completo



Lembrete

Dizer que um problema está em NP não é argumento para sua intratabilidade.

Dizer que ele é NP-completo/NP-difícil, sim.

Usando NP-completude para provar outras coisas (EXTRA - CURIOSIDADE)

Inexistência de algoritmos

Teorema

Não existe algoritmo eficiente que resolva o problema CAMINHOTODOPAR sobre $\langle D, w\rangle$ quando há ciclo negativo em D, a menos que $P = NP$.

Demonstração

Primeiro note que há uma redução polinomial de CAMINHOLONGOOPT para CAMINHOTODOPAR.
Seja $\langle D\rangle$ uma instância para CAMINHOLONGOOPT.
Faça $w(e) = -1$ para todo arco $e \in E(D)$.
Note que $\langle D, w\rangle$ é instância para CAMINHOTODOPAR.
Se P é o caminho de maior comprimento em D, então os vértices extremos de P têm a menor distância considerando a função w.
Da mesma forma, se u, v são pares de vértices com a menor distância considerando a função w, então um uv-caminho mínimo é um caminho de maior comprimento em D.

Inexistência de algoritmos

Teorema

Não existe algoritmo eficiente que resolva o problema CAMINHOTODOPAR sobre $\langle D, w\rangle$ quando há ciclo negativo em D, a menos que $P = NP$.

Demonstração (continuação)

Se D não tem ciclos, então encontrar o caminho mais longo em D pode ser feito em tempo polinomial. Logo, considere que D tem ciclos.
Suponha agora que existe um algoritmo eficiente ALG que é capaz de resolver o problema CAMINHOTODOPAR sobre qualquer digrafo que possua ciclos negativos.
Em particular, $ALG(D, w)$ encontrará um par x, y de vértices cuja distância é a menor dentre as distâncias de todos os pares de vértices de D.
Com isso, podemos usar ALG para resolver o problema CAMINHOLONGOOPT, que é NP-difícil.