Complexidade computacional

CLRS sec 34.1 e 34.2

Algumas questões

Por que alguns problemas parecem ser (computacionalmente) mais difíceis do que outros?

Algumas questões

Por que alguns problemas parecem ser (computacionalmente) mais difíceis do que outros?

Como podemos medir ou comprovar este diferente e aparentemente intrínseco grau de dificuldade?

Algumas questões

Por que alguns problemas parecem ser (computacionalmente) mais difíceis do que outros?

Como podemos medir ou comprovar este diferente e aparentemente intrínseco grau de dificuldade?

Quais problemas podem ser resolvidos por um algoritmo? Quais não podem? Por quê?

Algumas questões

Por que alguns problemas parecem ser (computacionalmente) mais difíceis do que outros?

Como podemos medir ou comprovar este diferente e aparentemente intrínseco grau de dificuldade?

Quais problemas podem ser resolvidos por um algoritmo? Quais não podem? Por quê?

Quais problemas podem ser resolvidos por um algoritmo eficiente? Quais não podem? Por quê?

Problema abstrato

Problema abstrato de decisão:

conjunto I de instâncias e

função $f : I \rightarrow \{\text{SIM}, \text{NÃO}\}$

Problema abstrato

Problema abstrato de decisão:
conjunto I de instâncias e
função $f : I \to \{\text{SIM}, \text{NÃO}\}$

Exemplo: Problema circuito hamiltoniano.

I: conjunto de todos os grafos.

Para todo G em I,

$$f(G) = \begin{cases} \text{SIM} & \text{se } G \text{ tem um circuito hamiltoniano} \\ \text{NÃO} & \text{caso contrário} \end{cases}$$

Problema abstrato

Problema abstrato de decisão:
conjunto I de instâncias e
função $f : I \to \{\text{SIM}, \text{NÃO}\}$

Exemplo: Problema circuito hamiltoniano.

I: conjunto de todos os grafos.

Para todo G em I,

$$f(G) = \begin{cases} \text{SIM} & \text{se } G \text{ tem um circuito hamiltoniano} \\ \text{NÃO} & \text{caso contrário} \end{cases}$$

O valor de $f(X)$ é
a resposta do problema para a instância X em I.

Problema concreto

Para um problema abstrato ser resolvido,
instâncias devem estar convenientemente codificadas.

Problema concreto

Para um problema abstrato ser resolvido,
instâncias devem estar convenientemente codificadas.

Problema concreto de decisão: problema abstrato de decisão em que as instâncias estão codificadas convenientemente em um alfabeto finito Σ.

Problema concreto

Para um problema abstrato ser resolvido,
instâncias devem estar convenientemente codificadas.

Problema concreto de decisão: problema abstrato de decisão em que as instâncias estão codificadas convenientemente em um alfabeto finito Σ.

Exemplo: Problema circuito hamiltoniano.

G dado pelas suas listas de adjacências.

Problema concreto

Para um problema abstrato ser resolvido,
instâncias devem estar convenientemente codificadas.

Problema concreto de decisão: problema abstrato de decisão em que as instâncias estão codificadas convenientemente em um alfabeto finito Σ.

Exemplo: Problema circuito hamiltoniano.

G dado pelas suas listas de adjacências.

Considere, sem perda de generalidade, que $\Sigma = \{0, 1\}$.

Problema concreto

Para um problema abstrato ser resolvido,
instâncias devem estar convenientemente codificadas.

Problema concreto de decisão: problema abstrato de decisão em que as instâncias estão codificadas convenientemente em um alfabeto finito Σ.

Exemplo: Problema circuito hamiltoniano.

G dado pelas suas listas de adjacências.

Considere, sem perda de generalidade, que $\Sigma = \{0, 1\}$.

Daqui para frente, todos os problemas são concretos.

Modelo de computação

Algoritmo:

programa escrito em uma linguagem como `C` ou `Pascal`.

Modelo de computação

Algoritmo:
programa escrito em uma linguagem como `C` ou `Pascal`.

Custo de uma operação:
proporcional ao número de bits dos operandos

Custo de um acesso à memória:
proporcional ao número de bits dos operandos.

Modelo de computação

Algoritmo:
programa escrito em uma linguagem como `C` ou `Pascal`.

Custo de uma operação:
proporcional ao número de bits dos operandos

Custo de um acesso à memória:
proporcional ao número de bits dos operandos.

(Isto é polinomialmente equivalente à máquina de Turing.)

Modelo de computação

Algoritmo:
programa escrito em uma linguagem como `C` ou `Pascal`.

Custo de uma operação:
proporcional ao número de bits dos operandos

Custo de um acesso à memória:
proporcional ao número de bits dos operandos.

(Isto é polinomialmente equivalente à máquina de Turing.)

Um algoritmo resolve ou decide um problema de decisão (I, f) se, para cada X em I, o algoritmo encontra a resposta para X, isto é, o algoritmo calcula $f(X)$.

Problemas indecidíveis

Existem problemas para os quais não existe um algoritmo: são os ditos problemas indecidíveis.

Problemas indecidíveis

Existem problemas para os quais não existe um algoritmo: são os ditos problemas indecidíveis.

Exemplo: Problema da parada

dado um programa e um arquivo de entrada para ele, decidir se tal programa pára ou não
quando executado com este arquivo de entrada.

Problemas indecidíveis

Existem problemas para os quais não existe um algoritmo: são os ditos problemas indecidíveis.

Exemplo: Problema da parada

dado um programa e um arquivo de entrada para ele, decidir se tal programa pára ou não
quando executado com este arquivo de entrada.

Não existe algoritmo que resolva o problema da parada.

A classe P

Um algoritmo resolve (ou decide) um problema de decisão (I, f) em tempo $\mathrm{O}(T(n))$ se, para cada n em N e cada X em I com $|X| = n$, o algoritmo encontra a resposta $f(X)$ para X em tempo $\mathrm{O}(T(n))$.

A classe P

Um algoritmo resolve (ou decide) um problema de decisão (I, f) em tempo $\mathrm{O}(T(n))$ se, para cada n em N e cada X em I com $|X| = n$, o algoritmo encontra a resposta $f(X)$ para X em tempo $\mathrm{O}(T(n))$.

Um problema de decisão é solúvel em tempo polinomial se existe algum k para o qual existe um algoritmo que resolve o problema em tempo $\mathrm{O}(n^k)$.

A classe P

Um algoritmo resolve (ou decide) um problema de decisão (I, f) em tempo $\mathrm{O}(T(n))$ se, para cada n em N e cada X em I com $|X| = n$, o algoritmo encontra a resposta $f(X)$ para X em tempo $\mathrm{O}(T(n))$.

Um problema de decisão é solúvel em tempo polinomial se existe algum k para o qual existe um algoritmo que resolve o problema em tempo $\mathrm{O}(n^k)$.

Tais problemas são ditos tratáveis.

A classe P

Um algoritmo resolve (ou decide) um problema de decisão (I, f) em tempo $\mathrm{O}(T(n))$ se, para cada n em N e cada X em I com $|X| = n$, o algoritmo encontra a resposta $f(X)$ para X em tempo $\mathrm{O}(T(n))$.

Um problema de decisão é solúvel em tempo polinomial se existe algum k para o qual existe um algoritmo que resolve o problema em tempo $\mathrm{O}(n^k)$.

Tais problemas são ditos tratáveis.

Classe de complexidade P:
conjunto dos problemas de decisão tratáveis
(isto é, que são solúveis em tempo polinomial)

A classe NP

Considere o problema circuito hamiltoniano.

A classe NP

Considere o problema circuito hamiltoniano.

Se a resposta para uma instância G for SIM, então existe um circuito hamiltoniano C no grafo.

A classe NP

Considere o problema circuito hamiltoniano.

Se a resposta para uma instância G for SIM,
então existe um circuito hamiltoniano C no grafo.

Dados G e C, é possível verificar em tempo polinomial em $|G|$ se C é de fato um circuito hamiltoniano em G ou não.

A classe NP

Considere o problema circuito hamiltoniano.

Se a resposta para uma instância G for SIM,
então existe um circuito hamiltoniano C no grafo.

Dados G e C, é possível verificar em tempo polinomial em $|G|$ se C é de fato um circuito hamiltoniano em G ou não.

Ou seja, temos como certificar eficientemente que a resposta SIM está correta.

A classe NP

Considere o problema circuito hamiltoniano.

Se a resposta para uma instância G for SIM,
então existe um circuito hamiltoniano C no grafo.

Dados G e C, é possível verificar em tempo polinomial em $|G|$ se C é de fato um circuito hamiltoniano em G ou não.

Ou seja, temos como certificar eficientemente que a resposta SIM está correta.

Conseguimos certificar eficientemente a resposta NÃO? Surpreendentemente, não se conhece nenhum método de certificação eficiente para a resposta NÃO no caso do problema circuito hamiltoniano.

A classe NP

Classe de complexidade NP: problemas para os quais a resposta SIM pode ser certificada e verificada em tempo polinomial.

A classe NP

Classe de complexidade NP: problemas para os quais a resposta SIM pode ser certificada e verificada em tempo polinomial.

Mais precisamente, um problema de decisão está em NP se existe um algoritmo A tal que

1. para qualquer instância X do problema com resposta SIM, existe um Y em Σ^* tal que $A(X,Y)$ devolve SIM;
2. para qualquer instância X do problema com resposta NÃO, para todo Y em Σ^*, $A(X,Y)$ devolve NÃO;
3. A consome tempo polinomial em $|X|$.

A classe NP

Classe de complexidade NP: problemas para os quais a resposta SIM pode ser certificada e verificada em tempo polinomial.

Mais precisamente, um problema de decisão está em NP se existe um algoritmo A tal que

1. para qualquer instância X do problema com resposta SIM, existe um Y em Σ^* tal que $A(X,Y)$ devolve SIM;
2. para qualquer instância X do problema com resposta NÃO, para todo Y em Σ^*, $A(X,Y)$ devolve NÃO;
3. A consome tempo polinomial em $|X|$.

Y é chamado de certificado para SIM da instância X.

A classe coNP

Complemento de um problema de decisão Q: problema de decisão obtido invertendo-se o SIM e o NÃO na resposta ao problema Q.

A classe coNP

Complemento de um problema de decisão Q: problema de decisão obtido invertendo-se o SIM e o NÃO na resposta ao problema Q.

Exemplo: Problema não-hamiltoniano

Resposta é SIM sempre que o grafo não tem um circuito hamiltoniano e NÃO caso contrário.

A classe coNP

Complemento de um problema de decisão Q: problema de decisão obtido invertendo-se o SIM e o NÃO na resposta ao problema Q.

Exemplo: Problema não-hamiltoniano

Resposta é SIM sempre que o grafo não tem um circuito hamiltoniano e NÃO caso contrário.

Classe de complexidade coNP: problemas de decisão que são complemento de problemas de decisão em NP.

A classe coNP

Complemento de um problema de decisão Q: problema de decisão obtido invertendo-se o SIM e o NÃO na resposta ao problema Q.

Exemplo: Problema não-hamiltoniano

Resposta é SIM sempre que o grafo não tem um circuito hamiltoniano e NÃO caso contrário.

Classe de complexidade coNP: problemas de decisão que são complemento de problemas de decisão em NP.

Problemas para os quais existe
um certificado (curto) para a resposta NÃO.

$\mathbf{P} \times \mathbf{NP} \times \mathbf{coNP}$

O problema circuito hamiltoniano é um exemplo de problema em NP enquanto que o problema não-hamiltoniano é um exemplo de problema em coNP.

P × NP × coNP

O problema circuito hamiltoniano é um exemplo de problema em NP enquanto que o problema não-hamiltoniano é um exemplo de problema em coNP.

Note ainda que P $\subseteq$ NP $\cap$ coNP.

$\mathbf{P} \times \mathbf{NP} \times \mathbf{coNP}$

O problema circuito hamiltoniano é um exemplo de problema em NP enquanto que o problema não-hamiltoniano é um exemplo de problema em coNP.

Note ainda que $\mathrm{P} \subseteq \mathrm{NP} \cap \mathrm{coNP}$.