Projeto e Análise de Algoritmos

Tempo polinomial
Verificação de tempo polinomial
Aula 25

Diane Castonguay
diane@inf.ufg.br

Instituto de Informática
Universidade Federal de Goiás

Tempo polinomial

Um algoritmo é ***eficiente***
se ele é de complexidade polinomial
no tamanho da entrada.

Ideia
tamanho da entrada : n
Complexidade do algoritmo: $O(n^a)$

Algoritmo de primalidade

```
Primo(n)
1. m ← SqrtInteiro(n)      // m = ⌊sqrt(n)⌋
2. para d ← 2 até m  faça
3.          se n div d = 0
4.               então retorna p "não é primo"
5.          fim-se
6. fim-para
```

Para uma solução melhor:
http://www.educ.fc.ul.pt/icm/icm98/icm12/Algoritmos.htm

Complexidade do algoritmo Primo

É $O(n^{1/2})$, mas o algoritmo Primo não é eficiente!

Porque???

Para implementar um numero inteiro usa-se o binario, isto quer dizer que o tamanho da entrada é $\Theta(\lg n)$ e não n.

$n^{1/2} = (2^{\lg n})^{1/2} = (\sqrt{2})^{\lg n}$ é exponencial em lg n!

Algoritmo de raiz quadrada inteira

/* $m = \lfloor sqrt(n) \rfloor$, $n > 1$*/
SqrtInteiro(n)
1. $m \leftarrow 1$
2. enquanto $(m+1)*(m+1) \leq n$ faça
3. $m \leftarrow m + 1$
4. fim-enquanto
5. retorna

Para uma solução melhor:
http://www2.fundao.pro.br/articles.asp?cod=151

Exemplo de algoritmos eficientes

Vimos varios exemplos ao longo da disciplina.

Há problemas pelos quais vimos
algoritmos não éficientes
e as suas soluções eficientes.

Um exemplo é os problemas que resolvemos com
a programação dinâmica.

Problema versus algoritmo

Tendo um problema e um algoritmo que o resolve,
se o algoritmo não é eficiente, isto quer dizer que
não existe um algoritmo eficiente que resolva o problema?

CLARO QUE NÃO!!!

Tipos de problemas

DECISÃO

Existe uma estrutura S que satisfaça uma propriedade P?

RESPOSTA: SIM ou NÃO

LOCALIZAÇÃO

Encontrar uma estrutura S que satisfaça uma propriedade P.

RESPOSTA: uma estrutura S

OPTIMIZAÇÃO

Encontrar uma estrutura S que satisfaça uma propriedade P e que seja melhor segundo algum critério C de medida.

RESPOSTA: uma estrutura S tal que não haja outra melhor

Definition de CLIQUE

Seja G = (V, E) um grafo não orientado.
Uma ***clique*** V' é um subconjunto de vértices de G ($V' \subseteq V$)
tal que para cada par de vértices u, v $\in$V'
temos uma aretsa (u, v) em G (i.e (u, v) $\in$ E).
O *tamanho de uma clique* V' é a quantia de vértices em V'.

Tipos de problemas: CLIQUE

DECISÃO

Dado um grafo G não orientado.
existe uma clique de tamanho maior ou igual a k?

LOCALIZAÇÃO

Dado um grafo G não orientado.
Encontre uma clique de tamanho maior ou igual a k.

OPTIMIZAÇÃO

Dado um grafo G não orientado.
Encontre uma clique de maoir tamanho.

Definição do Problema do caixeiro viajante

Seja G = (V, E) um grafo orientado com peso.

Um ***percurso*** é um ciclo simples
que passa por cada vertice do grafo.

O ***peso do percurso***
é a soma dos pesos das arestas do ciclo.

Tipo de problemas do caixeiro viajante

DECISÃO

Dado um grafo G orientado com peso.
existe um percurso de peso menor ou igual a k?

LOCALIZAÇÃO

Dado um grafo G orientado com peso.
Encontre um percurso de peso menor ou igual a k.

OPTIMIZAÇÃO

Dado um grafo G orientado com peso.
Encontre um percurso de menor peso.

CLASSE P

Um problema pertence a ***classe P*** (polinomial)
se existe um algoritmo eficiente que o resolva.

Para mostrar que um problema **é polinomial**
precisa exibir um algoritmo que o resolva
em tempo polinomial (no tamanho da entrada)

Para mostar que um problema **NÃO é polinomial**
precisa mostrar que nenhum algoritmo de complexidade
polinomial possa resolve-lo.

CERTIFICADO
ALGORITMO DE VERIFICAÇÃO

Vamos nos concentrar em problemas de decisão.
Muitos problemas de decisão podem ser resolvido através de um certificado.

Um ***certificado*** é uma estrutura S
que satisfaz a propriedade P.

Um ***algoritmo de verificação*** é um algoritmo
que prova se uma estrutura S satisfaz a propriedade P.

Verificação de um certificado
CLIQUE

E-CLIQUE(G, V')
1. para u, v ∈ V' faça
2. se (u, v) ∉ E
3. então retorna FALSE
4. fim-se
5. fim-para
6. retorna TRUE

Podemos implementar este algoritmo para seja $O(V^2)$.

CLASSE NP

Um problema de decisão pertence a ***classe NP***
(non-deterministic polinomial)
se a resposta *sim* do problema
tem certificado polinomial.

Isto é:
tem um algoritmo de verificação polinomial
da propriedade P por um certificado dado S.

Exemplo

CLIQUE é NP

SAT é NP

P $\subseteq$ NP

Exemplo: SAT

Entrada: uma fórmula booleana
Composição:
- variaveis booleanas
- conectivos booleanos
$\wedge$ (AND), $\vee$ (OR), $\neg$ (NOT),
$\rightarrow$ (implicação), $\leftrightarrow$ (Se e somente se)
- parênteses

Problemas relacionados a SAT
CIRCUIT-SAT, CNF-SAT, 3-CNF-SAT
FNC = forma normal conjunta : $\wedge$ (de $\vee$)
3-FNC = forma normal conjunta de 3 : $\wedge$ (3 de $\vee$)

Problema de decisão: SAT

Dado uma fórmula booleana, ele é capaz de satisfação?

SAT é NP

Para mostrar que SAT pertence a NP
mostramos que um certificado
(i.e uma atribuição satisfatória para a formula de entrada)
pode ser verificado em tempo polinomial.

O algoritmo de verificação
simplesmente subsitui cada variável
na fórmula por seu valor correspondente,
e então avalia a expressão.
Essa tarefa é realizável em tempo polinomial.
Se a expressão tem o valor 1,
a fórmula é capaz de satisfação.

CLASSE co-NP

Um problema de decisão pertence a ***classe co-NP***
se a resposta *não* do problema
tem certificado polinomial.

O grande questionamento!

$$P = NP???$$