ACH2043
INTRODUÇÃO À TEORIA DA COMPUTAÇÃO

Aula 25

Cap 7.2 – A classe P

Profa. Ariane Machado Lima
ariane.machado@usp.br

Cap 7.2 – A classe P

Tempo polinomial e exponencial

- Ex:
  - Máquina de tempo n^3 (tempo polinomial)
  - Máquina de tempo 2^n (tempo exponencial)
  - $n = 1000$:
  - $n^3 = 1$ bilhão
  - 2^n = número maior que o número de átomos do universo (aproximadamente 300 dígitos)

Tempo polinomial

- Todos os modelos computacionais **determinísticos** são polinomialmente equivalentes (um simula o outro com uma diferença de tempo polinomial)
- Problemas tratáveis na prática são polinomiais
- Em complexidade, muitas vezes só se quer saber se um problema é polinomial ou exponencial
  - Não importante o grau do polinômio
  - Reduzir complexidade de exponencial para polinomial muitas vezes permite atingir uma complexidade razoável para fins práticos

Tempo polinomial

- **Definição**: **P** é a classe de linguagens que são decidíveis em tempo polinomial sobre uma máquina de Turing determinística de uma fita. Em outras palavras,

$$P = \bigcup_k TIME(n^k).$$

- Importância:
  - P é invariante para todos os modelos de computação polinomialmente equivalentes à MT determinística de fita-única
  - P corresponde aproximadamente à classe de problemas que são realisticamente solúveis em um computador.

Analisando se um problema está em P

- Descreveremos ALGORITMOS (não considerando um modelo computacional específico)
  - Sem descrever fitas e movimentações de cabeça
- Um algoritmo é polinomial quando:
  - Cada estágio roda em tempo polinomial
  - Cada estágio é executado um número polinomial de vezes
  - Codificação e decodificação da entrada ocorrem em tempo polinomial

Exemplos de problemas em P

CAM pertence a P ?

- CAM = {<G,s,t> | G é um grafo direcionado que tem um caminho direcionado do nó s para o nó t}
- Codificação:
  - lista de nós e arestas
  - Matriz de adjacência
- Uma solução força-bruta (verifica todas as possibilidades):
  - Verifica todos os caminhos em potencial de comprimento no máximo m (m = número de nós)
  - Número de caminhos em potencial:

CAM pertence a P ?

- CAM = {<G,s,t> | G é um grafo direcionado que tem um caminho direcionado do nó s para o nó t}
- Codificação:
  - lista de nós e arestas
  - Matriz de adjacência
- Uma solução força-bruta (verifica todas as possibilidades):
  - Verifica todos os caminhos em potencial de comprimento no máximo m (m = número de nós)
  - Número de caminhos em potencial: aproximadamente m^m (exponencial!)

CAM pertence a P

- Alternativa: busca em grafo (ex: busca em largura):
  - Marcamos sucessivamente todos os nós em G que são atingíveis a partir de s por caminhos direcionados de comprimento 1, depois 2, depois 3, … até m.
  - Se existe um caminho direcionado de s a t, t ficará marcado

Teorema: CAM pertence a P

- Prova: O algoritmo M é polinomial:

M = “Sobre a entrada <G,s,t> onde G é um grafo direcionado com nós s e t:

1. Marque o nó s.
2. Repita até que nenhum nó adicional seja marcado:
   3. Faça uma varredura em todas as arestas de G. Se uma aresta (*a,b*) for encontrada indo de um nó marcado *a* para um nó não marcado *b*, marque o nó *b*
4. Se *t* estiver marcado, *aceite*. Caso contrário, *rejeite*.

-
  –
  –

Teorema: CAM pertence a P

- Prova: O algoritmo M é polinomial:

M = “Sobre a entrada <G,s,t> onde G é um grafo direcionado com nós s e t:

1. Marque o nó s.
2. Repita até que nenhum nó adicional seja marcado:
    3. Faça uma varredura em todas as arestas de G. Se uma aresta (*a,b*) for encontrada indo de um nó marcado *a* para um nó não marcado *b*, marque o nó *b*
4. Se *t* estiver marcado, *aceite*. Caso contrário, *rejeite*.

- Quantas vezes cada estágio executa?
    –
    –

Teorema: CAM pertence a P

- Prova: O algoritmo M é polinomial:

M = “Sobre a entrada <G,s,t> onde G é um grafo direcionado com nós s e t:

1. Marque o nó s.
2. Repita até que nenhum nó adicional seja marcado:
    3. Faça uma varredura em todas as arestas de G. Se uma aresta (*a,b*) for encontrada indo de um nó marcado *a* para um nó não marcado *b*, marque o nó *b*
4. Se *t* estiver marcado, *aceite*. Caso contrário, *rejeite*.

- Quantas vezes cada estágio é executado?
  - 1 e 4 são executados uma vez
  - 3 é executado no máximo m vezes (se cada vez marcar somente um nó)

Teorema: CAM pertence a P

- Prova: O algoritmo M é polinomial:

M = “Sobre a entrada <G,s,t> onde G é um grafo direcionado com nós s e t:

1. Marque o nó s.
2. Repita até que nenhum nó adicional seja marcado:
    3. Faça uma varredura em todas as arestas de G. Se uma aresta (*a,b*) for encontrada indo de um nó marcado *a* para um nó não marcado *b*, marque o nó *b*
4. Se *t* estiver marcado, *aceite*. Caso contrário, *rejeite*.

- Complexidade de cada estágio?
    –

Teorema: CAM pertence a P

- Prova: O algoritmo M é polinomial:

M = “Sobre a entrada <G,s,t> onde G é um grafo direcionado com nós s e t:

1. Marque o nó s.
2. Repita até que nenhum nó adicional seja marcado:
    3. Faça uma varredura em todas as arestas de G. Se uma aresta (*a,b*) for encontrada indo de um nó marcado *a* para um nó não marcado *b*, marque o nó *b*
4. Se *t* estiver marcado, *aceite*. Caso contrário, *rejeite*.

- Complexidade de cada estágio?
    – Polinomial

Teorema: CAM pertence a P

- Prova: O algoritmo M é polinomial:

M = “Sobre a entrada <G,s,t> onde G é um grafo direcionado com nós s e t:

1. Marque o nó s.
2. Repita até que nenhum nó adicional seja marcado:
    3. Faça uma varredura em todas as arestas de G. Se uma aresta (*a,b*) for encontrada indo de um nó marcado *a* para um nó não marcado *b*, marque o nó *b*
4. Se *t* estiver marcado, *aceite*. Caso contrário, *rejeite*.

- Codificação / decodificação de <G, s, t>
    - Polinomial

Teorema: CAM pertence a P

- Prova: O algoritmo M é polinomial:

M = “Sobre a entrada <G,s,t> onde G é um grafo direcionado com nós s e t:

1. Marque o nó s.
2. Repita até que nenhum nó adicional seja marcado:
    3. Faça uma varredura em todas as arestas de G. Se uma aresta (*a,b*) for encontrada indo de um nó marcado *a* para um nó não marcado *b*, marque o nó *b*
4. Se *t* estiver marcado, *aceite*. Caso contrário, *rejeite*.

Codificação / decodificação de <G, s, t>

– Polinomial

LOGO CAM Є P

PRIM-ES pertence a P

- PRIM-ES = {<x,y> | x e y são primos entre si}
- Primos entre si: 1 é o maior inteiro que divide AMBOS (ex: 10 e 21) – máximo divisor comum.
- Alternativa 1: testar todos os possíveis divisores
  - Problema: exponencial! (a magnitude de um número na base k cresce exponencialmente com o comprimento de sua representação)
- Alternativa 2:
  - Algoritmo euclidiano

PRIM-ES pertence a P

- Algoritmo euclidiano:

E = “Sobre a entrada <x,y>, onde x e y são números naturais em binário:

1. Repita até que y = 0:
2. $\quad x \leftarrow x \bmod y$
3. $\quad$ troque x e y
4. Dê como saída x.”

PRIM-ES pertence a P

- Algoritmo R que resolve PRIM-ES:

R = “Sobre a entrada <x,y>, onde x e y são números naturais em binário:

1. Rode E sobre <x,y>
2. Se o resultado for 1, *aceite*. Caso contrário, *rejeite*.”

PRIM-ES pertence a P

- Analisando E:

E = “Sobre a entrada <x,y>, onde x e y são números naturais em binário:

1. Repita até que y = 0:
2. $x \leftarrow x \bmod y$
3. troque x e y
4. Dê como saída x.”

- Execução do estágio 2 corta x para no mínimo sua metade:
  - Fim do estágio 2: $x < y$
  - Fim do estágio 3: $x > y$
  - Início do estágio 2: $x > y$
    - Se $x/2 >= y$ então $x \bmod y < y <= x/2$
    - Se $x/2 < y$, então $x \bmod y = x – y < x/2$

PRIM-ES pertence a P

- Analisando E:

E = “Sobre a entrada <x,y>, onde x e y são números naturais em binário:

1. Repita até que y = 0:
2. $x \leftarrow x \bmod y$
3. troque x e y
4. Dê como saída x.”

- Por causa do estágio 3, x e y são cortados no mínimo pela metade uma vez sim outra não.
  – Logo, estágios 2 e 3 são repetidos $\min(2 \log_2 x, 2 \log_2 y)$
  – Esses logs são proporcionais aos comprimentos das representações de x e y
  – Logo, estágios 2 e 3 são repetidos O(n) vezes

PRIM-ES pertence a P

- Analisando E:

E = “Sobre a entrada <x,y>, onde x e y são números naturais em binário:

1. Repita até que y = 0:
2. $\quad x \leftarrow x \bmod y$
3. $\quad$ troque x e y
4. Dê como saída x.”

- Cada estágio usa apenas tempo polinomial
- Tempo de execução total: polinomial

Toda Linguagem Livre de Contexto pertence a P

- Testar todas as possíveis derivações: exponencial
- Algoritmo **CYK** (para gramáticas na **forma normal de Chomsky**!)
- **Programação dinâmica**: uso de soluções de subproblemas menores para resolver subproblemas maiores (até chegar à solução do problema original)
- Tabela nxn:
  - i <= j : varíaveis que geram a subcadeia $w_i ... w_j$
  - Tratam-se tamanhos crescentes (começando de 1)

Toda Linguagem Livre de Contexto pertence a P

$D =$ "Sobre a entrada $w = w_1 \cdots w_n$:

1. Se $w = \varepsilon$ e $S \to \varepsilon$ for uma regra, *aceite*. [[trata o caso $w = \varepsilon$]]

2. Para $i = 1$ até n: [[examina cada subcadeia de comprimento 1]]

3. Para cada variável A:

4. Teste se $A \to \mathtt{b}$ é uma regra, onde $\mathtt{b} = w_i$.

5. Se for, coloque A em $tabela(i, i)$.

6. Para $l = 2$ até n: [[l é o comprimento da subcadeia]]

7. Para $i = 1$ até $n - l + 1$: [[i é a posição inicial da subcadeia]]

8. Faça $j = i + l - 1$, [[j é a posição final da subcadeia]]

9. Para $k = i$ até $j - 1$: [[k é a posição em que ocorre a divisão]]

10. Para cada regra $A \to BC$:

11. Se $tabela(i, k)$ contém B e $tabela(k + 1, j)$ contém C, ponha A em $tabela(i, j)$.

12. Se S estiver em $tabela(1, n)$, *aceite*. Caso contrário, *rejeite*."