CI1055: Algoritmos e Estruturas de Dados I

Profs. Drs. Marcos Castilho, Bruno Müller Jr, Carmem Hara

Departamento de Informática/UFPR

25 de agosto de 2020

Resumo

Aplicações de vetores: permutações (parte 3)

Objetivos da aula

- Resolver problemas que envolvem o uso de vetores
- Discutir eficiência dos algoritmos

Problemas com permutações

- (*) Testar se uma representação corresponde a uma permutação
- (*) Gerar aleatoriamente uma representação correspondente a uma permutação
- Determinar a ordem de uma permutação

Cálculo da ordem de uma permutação

$\boxed{\text{O que é a ordem de uma permutação?}}$

- Como $P(n)$ é uma permutação, então $P(P(n))$ também é
- Seja $P^2(n) = P(P(n))$
- Pode-se calcular, por exemplo, $P(P(1))$, isto é, $P^2(1)$

Exemplo

Considere a permutação:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 1 & 5 & 2 & 3 \end{pmatrix}$$

Então pode-se calcular:

- $P(P(1)) = P(4) = 2.$
- $P(P(2)) = P(1) = 4.$
- $P(P(3)) = P(5) = 3.$
- $P(P(4)) = P(2) = 1.$
- $P(P(5)) = P(3) = 5.$

Definição

$$\begin{cases} P^1(n) = P(n); \\ P^k(n) = P(P^{k-1}(n)), \quad k \geq 2. \end{cases}$$

Definição: permutação identidade

Quando $P(i) = i, \forall i$ a permutação recebe o nome de permutação identidade, ID.

$$ID = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix}$$

Definição: ordem da permutação

Seja $P(n)$ uma permutação sobre um conjunto de n elementos. Então existe um número natural k tal que $P^k = ID$.

Este número natural é chamado da *ordem* da permutação.

Exemplo

Considere a permutação:

$$P = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 1 & 5 & 2 & 3 \end{pmatrix}$$

Podemos calcular para valores pequenos de k como é P^k

Exemplo

$$P^1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 1 & 5 & 2 & 3 \end{pmatrix}$$

Exemplo

$$P^2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 3 & 1 & 5 \end{pmatrix}$$

Exemplo

$$P^3 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 5 & 4 & 3 \end{pmatrix}$$

Exemplo

$$P^4 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 1 & 3 & 2 & 5 \end{pmatrix}$$

Exemplo

$$P^5 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 5 & 1 & 3 \end{pmatrix}$$

Exemplo

$$P^6 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix}$$

Isto é, a ordem de P é 6

O problema computacional

- Queremos um algoritmo para calcular a ordem de uma permutação qualquer
- Este problema caiu na maratona de programação da ACM
- O algoritmo que gera todas as k permutações é ineficiente, teria que simular todo o processo e a cada etapa testar se o resultado é a permutação *ID*
- Existe um método melhor

Ideia do algoritmo

- Cada elemento $P(i) = x$ do conjunto retorna à posição i ciclicamente de *cont* em *cont* permutações
- Ou seja, $P^{cont}(i) = x, P^{2 \times cont}(i) = x, \ldots$
- O mesmo ocorre para todos elementos do conjunto, mas cada um possui um ciclo (valor de *cont*) próprio

Exemplo

Considere a permutação:

$$P = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 1 & 5 & 2 & 3 \end{pmatrix}$$

- Para o índice 1, temos que $P^3(1) = 1$
- Isto quer dizer que para todo múltiplo de 3 (a cada 3 iterações) $P^{3k}(1) = 1$

Exemplo

- Para este exemplo, temos que:
  - $P^3(1) = 1$
  - $P^3(2) = 1$
  - $P^3(4) = 1$
  - $P^2(3) = 1$
  - $P^2(5) = 1$

Conclusão

- Pode-se concluir que a permutação *ID* ocorrerá na iteração que é o mínimo múltiplo comum (MMC) entre o número que provoca repetição entre todos os índices.
- Observação:
  - $MMC(x_1, x_2, \ldots, x_n) = MMC(x_1, MMC(x_2, \ldots, x_n))$

Algoritmo para o MMC

- Não existe algoritmo eficiente para cálculo do MMC
- Mas:
  - Existe para o MDC!

$$MDC(a, b) = \frac{a \times b}{MMC(a, b)}$$

```
function ordem_permutacao (var v: vetor_i; n: integer): int64;
var mmc, cont: int64;
    p, i: integer;
begin
    mmc := 1;
    for i := 1 to n do
    begin
        cont := 1;
        p := i;
        while (v[p] <> i) do
        begin
            cont:= cont + 1;
            p := v[p];
        end;
        mmc := mmc * cont div mdc(mmc, cont);
    end;
    ordem_permutacao:= mmc;
end;
```

Em resumo. . .

- Aprendemos a usar vetores como estrutura de dados para resolver um problema matemático
- Relembramos que existem algoritmos melhores do que outros, sob algum aspecto
- Fizemos um breve estudo sobre os algoritmos implementados
- Estudamos um problema de maratona de programação!

Fim do tópico

- este material está no livro no capítulo 9, seção 9.7.1

Licença

- Slides feitos em LaTeX usando beamer
- Licença

  Creative Commons Atribuição-Uso Não-Comercial-Vedada a Criação de Obras Derivadas 2.5 Brasil License.`http://creativecommons.org/licenses/by-nc-nd/2.5/br/`

  Creative Commons Atribuição-Uso Não-Comercial-Vedada a Criação de Obras Derivadas 2.5 Brasil License.`http://creativecommons.org/licenses/by-nc-nd/2.5/br/`