MAC323 - Estruturas de Dados

Notas de Aula - Ordenação Topológica

Siang - 2006 - Material baseado no livro de Knuth - *The Art of Computer Programming - Volume I.*

Ordem parcial

Uma ordem parcial de um conjunto S é uma relação entre os objetos de S, indicada pelo s¡mbolo $\preceq$ (leia-se "precede ou igual"), satisfazendo as seguintes propriedades para quaisquer objetos x, y, z de S (não necessariamente distintos):

(i) Transitividade: se $x \preceq y$ e $y \preceq z$ então $x \preceq z$.

(ii) Anti-simétrica: se $x \preceq y$ e $y \preceq x$ então $x = y$.

(iii) Reflexividade: $x \preceq x$.

Se $x \preceq y$ e x not $\preceq y$, então escreveremos $x \prec y$ e diremos que "x precede y".

De (i), (ii) e (iii) temos:

(i') se $x \prec y$ e $y \prec z$ então $x \prec z$.

(ii') se $x \prec y$ então y not $\prec x$.

(iii') x not $\prec x$.

Exemplos de ordens parciais:

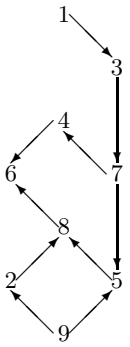
1. Relação $\leq$ (menor ou igual) entre números.
2. Relação $\subseteq$ (contido em) entre conjuntos.
3. Relação "x deve ser executado antes de y" em um conjunto de atividades.

Uma representação em diagrama

Vamos representar $x \prec y$ por $x \rightarrow y$. Assim as relações:

$9 \prec 2 \quad 4 \prec 6$
$3 \prec 7 \quad 1 \prec 3$
$7 \prec 5 \quad 7 \prec 4$
$5 \prec 8 \quad 9 \prec 5$
$8 \prec 6 \quad 2 \prec 8$

são representadas pelo diagrama.



A propriedade (ii) significa que não existem ciclos fechados.

O problema da ordenação topológica

Dadas as relações de uma ordem parcial, uma ordenação topológica é uma sequência

$$a_1 a_2 \ldots a_n$$

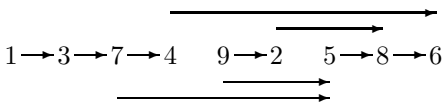
tal que para $a_j \prec a_k$, temos $j < k$.

Isto é, se um elemento a_j precede a_k, ele irá aparecer antes de a_k na ordenação topológica.

Um exemplo de uma ordenação topológica do exemplo acima é

1 3 7 4 9 2 5 8 6

Usando o diagrama,



Todas as "flechas" apontam para a direita.

Um algoritmo de ordenação topológica

Sejam n objetos numerados de 1 a n. Seja dado o valor n como a primeira entrada. Os dados seguintes de entrada são pares da forma

$$j \quad k$$

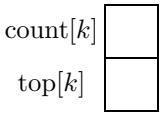
onde cada par significando $j \prec k$.

O último par contém

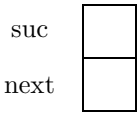
$$0 \quad 0$$

indicando o fim dos dados.

Usamos uma tabela sequencial $x[1], x[2], \ldots, x[n]$ onde cada $x[k]$ tem a forma

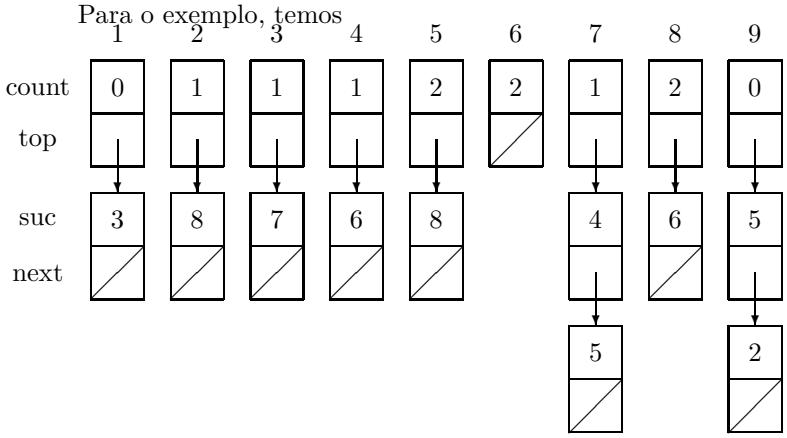


count[k] contém o número de predecessores diretos do objeto k, isto é, o número de pares ($j \prec k$) que aparecem na entrada. top[k] contém um apontador a uma lista de sucessores diretos do objeto k. Cada elemento dessa lista tem dois campos chamados suc e next.



O campo suc indica o sucessor direto e next aponta para o outro sucessor, se houver.

Para o exemplo, temos



O algoritmo de ordenação topológica fica então

I. [inicialização]

```
read(n); AindaSobrou := n;
for i := 1 step 1 until n do
    begin
        count[i] := 0;
        top[i] := nil
    end;
```

II. [Leitura dos pares $j \prec k$ e construção das listas]

```
read (j,k);
while j not = 0 do
    begin
        count[k] := count[k] + 1;
        extrai(P);
        suc(P) := k;
        next(P) := top[j];
        top[j] := P;
        read (j,k)
     end;
```

III. [Coloca todos os nós com count = 0 numa fila para facilitar a procura pelos elementos com count nulo. Supomos existência das rotinas InsereFila(i) e RemoveFila(i) que significam, respectivamente, inserção do elemento i na fila e remove um elemento i da fila.]

```
Inicialize a fila como vazia.
for i := 1 step 1 until n do
    if count[i] = 0 then
       InsereFila(i)
```

IV. [Produção de uma ordenação topológica.]

```
while Fila not vazia do
    begin
        RemoveFila(i);
        Imprima i como resposta;
        AindaSobrou := AindaSobrou - 1;
        P :=  top[i];
        while P not = nil do
            begin
                count[suc(P)] := count[suc(P)] - 1;
                if count[suc(P)] = 0 then
                   InsereFila(suc(P))
                P := next(P)
            end;
     end;
```

V. [Verificação final.]

```
if AindaSobrou > 0 then
    error;
    {existe um ciclo fechado com AindaSobrou elementos, não dá
     para produzir uma ordenação topológica.}
```