Proiectarea algoritmilor

Tema de casă nr. 1

Cuprins

Sortare topologică
- Considerații generale
- Algoritm BFS
- Exemplu
- Complexitatea timp

Sarcini de lucru și barem de notare
Bibliografie

Sortare topologică - considerații generale

- Se aplică la secvențe cu elemente din mulțimi parțial ordonale.
- Exemplu de relație de ordine este parțială: $a_1 < a_0, a_1 < a_2 < a_3$.
- Problema constă în a crea o listă liniară care să fie compatibilă cu relația de ordine, adică, dacă $a_i < a_j$, atunci a_i va precede pe a_j în lista finală.
- Pentru exemplul nostru, lista liniară finală va putea fi (a_1, a_0, a_2, a_3), sau (a_1, a_2, a_0, a_3), sau (a_1, a_2, a_3, a_0).
- **Definiție:** Fie $(S, \leq)$ o mulțime parțial ordonată finită și $a = (a_0, a_1, \ldots, a_{n-1})$ o liniarizare a sa. Spunem că secvența a este *sortată topologic*, dacă $\forall i, j : a_i < a_j \Rightarrow i < j$.

Mulțimi parțial ordonate finite și digrafuri aciclice

- Există o legătură strânsă între mulțimile parțial ordonate finite și digrafurile aciclice (digrafuri fără circuite numite pe scurt dag-uri).
- Un *graf* este o pereche $G = (V, E)$, unde V este o mulțime ale cărei elemente sunt numite *vârfuri*, iar E este o mulțime de perechi neordonate $\{u, v\}$ de vârfuri, numite *muchii*.
- Un *digraf* este o pereche $D = (V, A)$, unde V este o mulțime de vârfuri, iar A este o mulțime de perechi ordonate (u, v) de vârfuri, numite arce.
- Orice mulțime parțial ordonată $(S, \leq)$ definește un dag $D = (S, A)$, unde există arc de la a la b, dacă $a < b$ și nu există $c \in S$ cu proprietatea $a < c < b$.
- Reciproc, orice dag $D = (V, A)$ definește o relație de ordine parțială $\leq$ peste V, dată prin: $u \leq v$, dacă există un drum de lungime ≥ 0 de la u la v.
- De fapt, $\leq$ este închiderea reflexivă și tranzitivă a lui A (se mai notează $\leq = A^*$).
- Sortarea topologică a unui dag constă într-o listă liniară a vârfurilor astfel încât dacă există arc de la u la v, atunci u precede pe v în listă, pentru oricare două vârfuri u și v.
- Vârfurile care candidează pentru primul loc în lista sortată topologic au proprietatea că nu există arce incidente spre interior (care sosesc în acel vârf) și se numesc *surse*.

Reprezentarea digrafurilor prin liste de adiacență

- Reprezentarea prin liste de adiacență exterioară:
  - Digraful D este reprezentat printr-o structură asemănătoare cu cea de la matricele de adiacență.
  - Matricea de adiacență este înlocuită cu un tablou unidimensional de n liste liniare, implementate prin liste simplu înlănțuite și notate cu `D.a[i]` pentru $i = 0, \ldots, n-1$.
  - Lista `D.a[i]` conține vârfurile destinatare ale arcelor care pleacă din i (= lista de adiacență exterioară).
- Reprezentarea prin liste de adiacență interioară:
  - Lista `D.a[i]` conține vârfurile surse ale arcelor care sosesc în i (= lista de adiacență interioară).

Exemplu de reprezentare a digrafurilor prin liste de adiacenţă exterioară

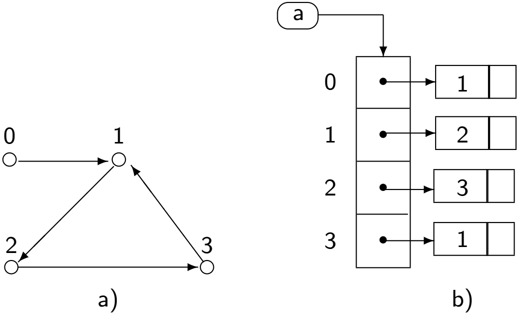


Figura 1 : Digraf reprezentat prin liste de adiacenţă exterioară înlănţuite

Sortare topologică - metoda BFS

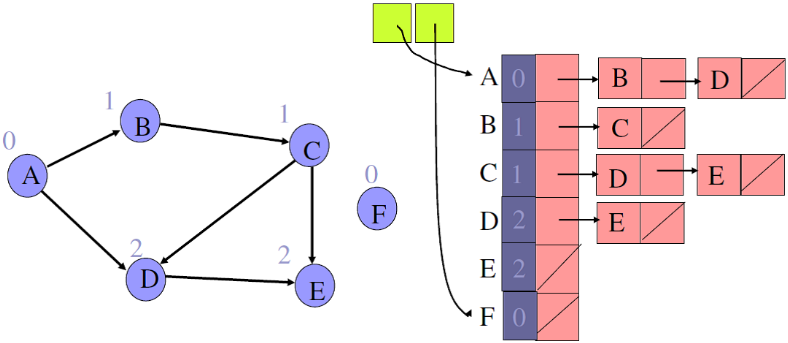
- Presupunem că pentru dag-ul D sunt create atât listele de adiacență interioară, cât și cele de adiacență exterioară.
- Listele de adiacență interioară vor fi utilizate la determinarea vârfurilor sursă (vârfuri fără predecesori); acestea au listele de adiacență interioară vide.
- Descrierea algoritmului:
    1. Inițializează coada cu vârfurile sursă.
    2. Extrage un vârf u din coadă pe care-l adaugă la lista sortată parțial.
    3. Elimină din reprezentarea (acum parțială) a lui D vârful u și toate arcele (u, v).
    4. Dacă pentru un astfel de arc lista de adiacență interioară a vârfului v devine vidă, atunci v va fi adăugat la coadă.
    5. Repetă pașii 2-4 până când coada devine vidă.

Sortare topologică - algoritm BFS

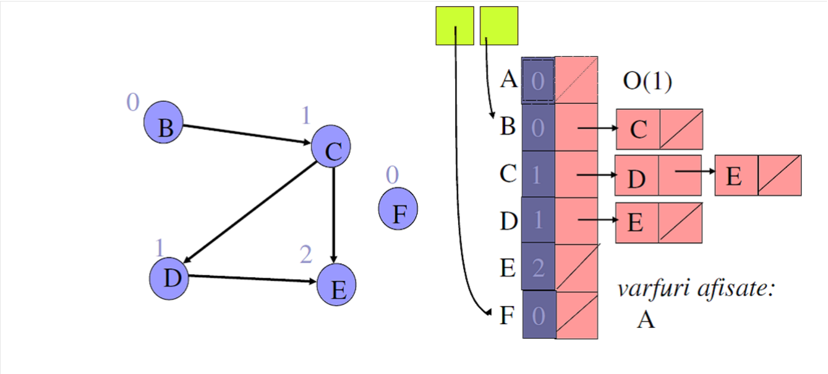
- Extindem structura D, care reprezintă digraful D, cu tabloul np[1..n].
- D.np[u] conține numărul predecesorilor vârfului u.
- L este lista care conține varfurile digrafului D în ordine topologică.

```
procedure sortareTopologicaBFS(D,np)
   coadaVida(C) //initializeaza coada C
   // insereaza in C varfurile fara predecesori
   for u ← 0 to D.n-1 do
       if D.np[u]=0 then insereaza(C,u)
   // construieste lista varfurilor (afiseaza) in ordine topologica
   for k ← 0 to D.n-1 do
       if esteVida(C)
          then return ("Graful contine cicluri")
       u ← elimina(C)
       inserează(L, u) // inserarea se face la sfarsitul listei
       p ← D.a[u]
       while p≠NULL do
          v ← p->elt //v este un succesor imedial al lui u
          D.np[v] ← D.np[v]-1
          if D.np[v]=0
             then insereaza(C,v)
          p ← p->succ
end
```

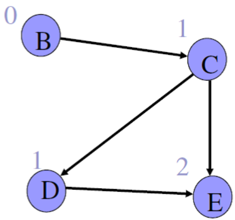
Exemplu de execuție a algoritmului de sortare toplogică prin metoda BFS

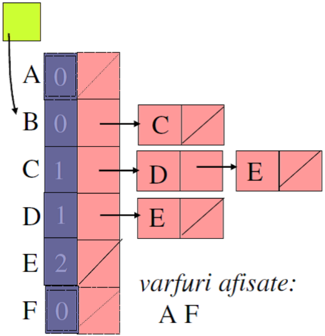


Exemplu de execuție a algoritmului de sortare toplogică prin metoda BFS

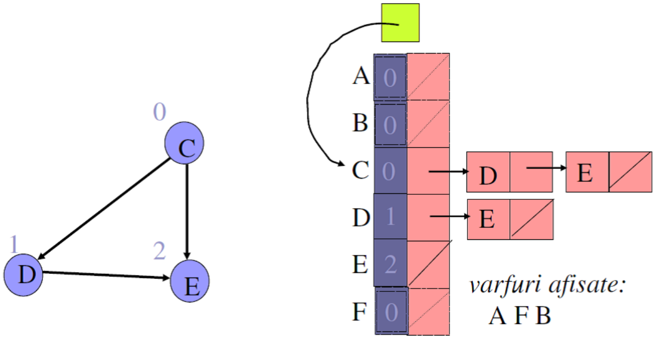


Exemplu de execuție a algoritmului de sortare toplogică prin metoda BFS

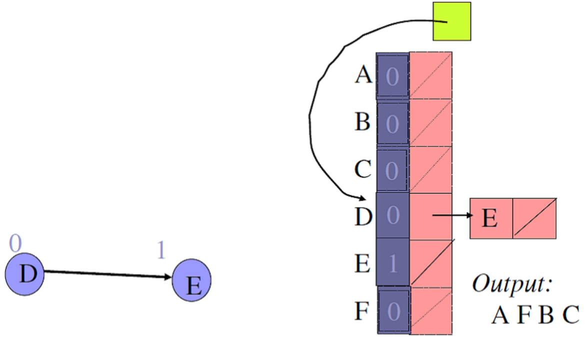




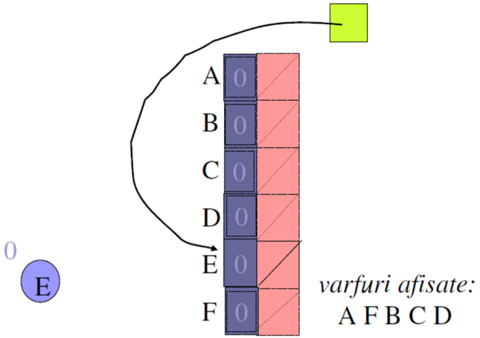
Exemplu de execuție a algoritmului de sortare toplogică prin metoda BFS



Exemplu de execuție a algoritmului de sortare toplogică prin metoda BFS

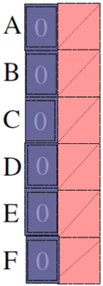


Exemplu de execuție a algoritmului de sortare toplogică prin metoda BFS



Exemplu de execuție a algoritmului de sortare toplogică prin metoda BFS

FINAL!



varfuri afisate:
A F B C D E

Complexitatea timp

- Compusă din timpul pentru:
  - identificarea vârfurilor fără predecesori: $O(n)$
  - ștergerea muchiilor: $O(m)$
  - ạfisarea vârfurilor: $O(n)$
- Timp total : $O(n+m)$

Sarcini de lucru și barem de notare

Sarcini de lucru:

1. Scrieți un program C/C++ care sortează topologic vârfurile unui digraf aciclic $D = (V, A)$.

Barem de notare:

1. Construirea listelor de adiacență: 3p
2. Implementarea algoritmului de sortare topologică : 6p
3. Baza: 1p

Bibliografie

Lucanu, D. și Craus, M., *Proiectarea algoritmilor*, Editura Polirom, 2008.