

Tema Nr. 10: Căutare în adâncime (DFS)

Timp Alocat: 2 ore

Implementare

Se cere implementarea corectă și eficientă a algoritmului de căutare în adâncime (Depth-First Search - DFS) (*Capitolul 22.3¹*). Pentru reprezentarea grafurilor, va trebui să folosești liste de adiacență. De asemenea, va trebui să:

- Implementarea algoritmului Tarjan pentru componente tare conexe
- Implementezi sortarea topologică (*Capitolul 22.4¹*)

Cerințe

1. DFS (5p)

Demonstrați corectitudinea algoritmului pe un graf de dimensiune mică:

- afișați graful inițial (liste de adiacență)
- afișați arborele rezultat în urma DFS

2. Sortare topologică (1p)

Demonstrați corectitudinea algoritmului pe un graf de dimensiune mică:

- afișați graful inițial (liste de adiacență)
- afișați listă de noduri sortate topologic (dacă are / dacă nu are de ce nu are?)

3. Tarjan (2p)

Demonstrați corectitudinea algoritmului pe un graf de dimensiune mică:

- afișați graful inițial (liste de adiacență)
- afișați componentele puternic conexe ale grafului

4. Analiza performanței pentru DFS (2p)

! Înainte de a începe să lucrați la partea de evaluare, asigurați-vă că aveți un algoritm corect!

Cum timpul de execuție al algoritmului DFS variază în funcție de numărul de vârfuri ($|V|$) și de numărul de muchii ($|E|$) aveți de făcut următoarele analize:

1. Fixați $|V|=100$ și variați $|E|$ între 1000 și 4500 cu un pas de 100. Generați pentru fiecare caz un graf aleator și asigurați-vă că nu generați aceeași muchie de 2 ori. Execută DFS pentru fiecare graf generat și numără operațiile efectuate. Apoi construiește graficul cu variația numărului de operații în funcție de $|E|$;
2. Fixați $|E|=4500$ și variați $|V|$ între 100 și 200 cu un pas de 10. Repetă procedura de mai sus și construiește graficul cu variația numărului de operații în funcție de $|V|$.

¹ Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein. *Introduction to Algorithms*