Część programistyczna

Należy zaimplementować dwa algorytmy sortowania topologicznego wierzchołków skierowanego grafu acyklicznego (DAG):

1. Algorytm Tarjana oparty na trawersowaniu grafu metodą przeszukiwania w głąb (DFS);
2. Algorytm Kahna oparty na iteracyjnym usuwaniu wierzchołków o zerowym stopniu wejściowym.

Oba algorytmy należy zaimplementować w taki sposób, aby przetwarzały grafy zapisane w każdej z trzech reprezentacji maszynowych grafu: macierz sąsiedztwa (ms), lista następników (ln) oraz lista krawędzi (lk). W konsekwencji należy zaimplementować następujące wersje algorytmów: Tarjan-ms, Tarjan-ln, Tarjan-lk, Kahn-ms, Kahn-ln, Kahn-lk. Każda wersja algorytmu ma operować wyłącznie na wskazanej reprezentacji grafu, tj. wszystkie operacje (m.in. trawersowanie grafu, wyznaczanie sąsiadów, obliczanie stopni wejściowych, usuwanie wierzchołków i krawędzi) muszą być realizowane bezpośrednio na tej strukturze danych. Niedozwolone jest wspomaganie działania algorytmu dodatkowymi reprezentacjami grafu (np. tworzenie list następników na podstawie macierzy sąsiedztwa lub odwrotnie). W szczególności nie należy przechowywać tej samej informacji o grafie w więcej niż jednej reprezentacji. Celem zadania jest bowiem porównanie efektywności algorytmów w zależności od zastosowanej reprezentacji grafu, dlatego każda implementacja powinna w pełni wykorzystywać wyłącznie swoją natywną strukturę danych.

Program powinien:

1. Generować losowy DAG lub wczytywać graf z pliku tekstowego (pierwsza linia: liczba wierzchołków n i liczba krawędzi m, kolejne linie: pary wierzchołków (u, v), wierzchołki numerowane od 1).
2. Wykrywać cykle: w przypadku wykrycia cyklu należy przerwać działanie algorytmu, wyświetlić komunikat o braku możliwości sortowania oraz wylistować wierzchołki tworzące wykryty cykl.
3. W trybie demonstracyjnym umożliwiać wybór reprezentacji grafu, wybór algorytmu oraz wybór wierzchołka startowego dla algorytmu Tarjana.
4. Dla każdego uruchomienia algorytmu podawać czas wykonania.
5. W trybie demonstracyjnym (dla małych grafów, $n \leq 12$) wyświetlać: graf w wybranej reprezentacji maszynowej, dla algorytmu Tarjana – moment odwiedzenia i przetworzenia każdego wierzchołka, dla algorytmu Kahna – stopnie wejściowe wierzchołków w kolejnych krokach działania algorytmu Kahna.

Część eksperymentalna

Generowanie danych: dla każdej liczby wierzchołków n należy generować acykliczne grafy skierowane o zadanym poziomie nasycenia s, rozumianym jako stosunek liczby krawędzi do maksymalnej liczby krawędzi w DAG o n wierzchołkach. W eksperymentach badane są grafy o nasyceniach $s \in [10\%, 90\%]$.

Dwa niezależne eksperymenty:

1. Eksperyment 1 – liniowy wzrost n. Należy wybrać 10–15 różnych wartości n rosnących liniowo, np.: 1000, 2000, 3000, 4000, … Zakres należy dobrać tak, aby możliwe było zaobserwowanie wzrostu czasu obliczeń wraz ze wzrostem n oraz aby pomiary były możliwe do wykonania w rozsądnym czasie. Eksperyment należy przeprowadzić dla trzech poziomów nasycenia grafu: niskie nasycenie (ok. 10%), średnie nasycenie (ok. 50%), wysokie nasycenie (ok. 90%).
2. Eksperyment 2 – liniowy wzrost s. Należy eksperymentalnie dobrać stałą wartość n, tak aby możliwe było zaobserwowanie różnic w czasie obliczeń oraz wszystkie pomiary były wykonalne w rozsądnym czasie. Następnie należy przeprowadzić eksperyment dla 9 wartości nasycenia s rosnących liniowo od 10% do 90% (z krokiem co 10).

Procedura pomiarowa (dla obu eksperymentów): dla każdej kombinacji algorytmu, reprezentacji grafu, liczby wierzchołków n oraz nasycenia s, należy wygenerować 10 niezależnych grafów i zmierzyć czas

sortowania topologicznego. Następnie należy obliczyć średni czas sortowania (punkt na wykresie) oraz odchylenie standardowe. Każdy punkt pomiarowy musi być średnią z 10 uruchomień algorytmu. Wszystkie wyniki należy zapisać w tabelach.

Sprawozdanie

Sprawozdanie powinno zawierać:

1. Wykresy dla eksperymentu 1 (liniowy wzrost liczby wierzchołków *n*). Należy przygotować 3 wykresy funkcji *t*=f(*n*), osobno dla każdej reprezentacji grafu: macierz sąsiedztwa (ms), lista następników (ln), lista krawędzi (lk). Na każdym wykresie należy przedstawić zależność średniego czasu obliczeń *t* (oś Y) od liczby wierzchołków *n* (oś X) i wyrysować 6 krzywych: algorytm Tarjana dla trzech poziomów nasycenia (niskie, średnie, wysokie), algorytm Kahna dla trzech poziomów nasycenia (niskie, średnie, wysokie). Dla każdego punktu pomiarowego należy zaznaczyć odchylenie standardowe (np. w postaci słupka błędu). Celem jest porównanie działania obu algorytmów w ramach tej samej reprezentacji grafu oraz analiza wpływu nasycenia grafu na czas obliczeń.

2. Wykresy dla eksperymentu 2 (liniowy wzrost nasycenia grafu *s*). Należy przygotować 2 wykresy funkcji *t*=f(*s*), osobno dla każdego algorytmu: algorytm Tarjana, algorytm Kahna. Na każdym wykresie należy przedstawić zależność średniego czasu obliczeń *t* (oś Y) od nasycenia grafu *s* (oś X) i umieścić na nim 3 krzywe – po jednej dla każdej reprezentacji grafu (ms, ln, lk). Dla każdego punktu pomiarowego należy zaznaczyć odchylenie standardowe (np. w postaci słupka błędu). Celem jest analiza wpływu gęstości grafu na efektywność algorytmów oraz porównanie reprezentacji grafu dla ustalonego rozmiaru danych.

3. Analizę wyników obu eksperymentów. Należy przeanalizować wyniki obu eksperymentów dla obu algorytmów i wszystkich reprezentacji grafu i odpowiedzieć na pytania:
   - Która reprezentacja maszynowa grafu jest najlepsza dla rzadkich grafów?
   - Która reprezentacja maszynowa grafu traci efektywność dla gęstych grafów?
   - Czy macierz sąsiedztwa jest efektywna w algorytmie Tarjana (i dlaczego tak lub dlaczego nie)?
   - Czy lista następników jest korzystna dla algorytmu Kahna (i dlaczego tak lub dlaczego nie)?
   - Jak koszt wyznaczania stopni wejściowych wpływa na algorytm Kahna?
   - Czy różnice między reprezentacjami rosną wraz z liczbą wierzchołków lub gęstością grafu?

4. Analizę złożoności obliczeniowej. Dla każdej reprezentacji maszynowej grafu należy oszacować złożoność czasową sortowania topologicznego algorytmem Tarjana i algorytmem Kahna. Należy porównać wyniki teoretyczne z wynikami eksperymentalnymi i wskazać ewentualne rozbieżności.

5. Wnioski końcowe. W podsumowaniu należy wskazać najbardziej efektywną reprezentację dla każdego algorytmu, określić kompromisy pamięć vs czas, wskazać przypadki, w których wybór reprezentacji ma kluczowe znaczenie.