Zadanie projektowe nr 2

Badanie efektywności algorytmów grafowych w zależności od rozmiaru instancji oraz sposobu reprezentacji grafu w pamięci komputera

Należy zaimplementować oraz dokonać pomiaru czasu działania wybranych algorytmów grafowych rozwiązujących następujące problemy:

a) wyznaczanie minimalnego drzewa rozpinającego (MST) – algorytm Kruskala oraz algorytm Prima,
b) wyznaczanie najkrótszej ścieżki w grafie – algorytm Dijkstry oraz algorytm Bellmana-Forda,
c) wyznaczanie maksymalnego przepływu – algorytm Forda-Fulkersona (tylko na ocenę 5.0 i 5.5).

Algorytmy te należy zaimplementować dla obu poniższych reprezentacji grafu w pamięci komputera:

- reprezentacja macierzowa (macierz wag - macierz sąsiedztwa w której zamiast wartości 0/1 wpisane są wagi krawędzi, umowna wartość ∞ oznacza brak krawędzi),
- reprezentacja listowa (połączone listy sąsiadów).

Należy przyjąć następujące założenia:

- wszystkie struktury danych powinny być alokowane dynamicznie,
- waga (przepustowość) krawędzi jest liczbą całkowitą,
- po zaimplementowaniu każdego z algorytmów dla obu reprezentacji należy dokonać pomiaru czasu działania algorytmów w zależności od rozmiaru grafu oraz jego gęstości (stosunku liczby krawędzi do maksymalnej możliwej liczby krawędzi przy danej liczbie wierzchołków). Badania należy wykonać dla 5 różnych (reprezentatywnych) liczb wierzchołków oraz następujących gęstości grafu: 25%, 50%, 75% oraz 99%. Dla każdego zestawu: reprezentacja, liczba wierzchołków i gęstość należy wygenerować po 100 losowych instancji, zaś w sprawozdaniu umieścić wyniki uśrednione,
- dodatkową funkcją programu musi być możliwość sprawdzenia poprawności zbudowanej struktury i zaimplementowanych operacji (szerzej na ten temat w dalszej części),
- sposoby dokładnego pomiaru czasu w systemie Windows podano na stronie www:

  http://staff.iiar.pwr.wroc.pl/antoni.sterna/sdizo/SDiZO_time.pdf

- dopuszczalnymi językami programowania są języki kompilowane do kodu natywnego (na przykład C, C++), a nie interpretowane lub uruchamiane na maszynach wirtualnych (np. JAVA,.NET, Phyton),
- używanie okienek nie jest konieczne i nie wpływa na ocenę (wystarczy wersja konsolowa),
- nie wolno korzystać z gotowych bibliotek takich jak STL, Boost lub innych przy implementacji głównych algorytmów (wyjątki zapisane przy ocenianiu). Można ich użyć

tylko w operacjach pomocniczych (przygotowanie danych testowych, wyświetlanie). Wszystkie algorytmy i struktury muszą być zaimplementowane samodzielnie (nie kopiować gotowych rozwiązań),

- implementacja powinna być wykonana w postaci jednego programu,
- kod źródłowy powinien być odpowiednio komentowany.

Sprawdzenie poprawności zbudowanej struktury/operacji obejmuje:

- wczytanie struktury grafu z pliku tekstowego (powinna być możliwość podania nazwy pliku). Plik zawiera informacje ogólne o grafie i opis poszczególnych krawędzi w następującym formacie:
    a. w pierwszej linii zapisana jest informacja o grafie (jednolita dla wszystkich problemów): liczba krawędzi, liczba wierzchołków, wierzchołek początkowy, wierzchołek końcowy (rozdzielone białymi znakami). W przypadku MST wartość dwóch ostatnich parametrów jest nieistotna, dla najkrótszej ścieżki zignorowana powinna być wartość wierzchołka końcowego,
    b. wierzchołki numerowane są w sposób ciągły od zera,
    c. w kolejnych liniach znajduje się opis krawędzi (każda krawędź w osobnej linii) w postaci trzech liczb rozdzielonych białymi znakami (wierzchołek początkowy, wierzchołek końcowy oraz waga/przepustowość),
    d. dla problemu MST krawędzie traktowane są jako nieskierowane, natomiast dla algorytmów najkrótszej ścieżki i maksymalnego przepływu jako skierowane,
- wyświetlenie wczytanego grafu w formie reprezentacji macierzowej i listowej,
- uruchomienie algorytmu dla obu reprezentacji i wyświetlenie wyników na ekranie.

Poniższe operacje należy zrealizować w formie menu (osobno dla każdego problemu):

1. Wczytaj graf z pliku,
2. Wyświetl graf macierzowo i listowo,
3. Algorytm 1 (np. Kruskala) macierzowo i listowo z wyświetleniem wyników,
4. Algorytm 2 (np. Prima) macierzowo i listowo z wyświetleniem wyników.

Wyświetlanie wyników:

a) w przypadku MST wyświetlić listę krawędzi drzewa rozpinającego oraz sumę wag tych krawędzi,

b) dla problemu najkrótszej drogi dla każdego wierzchołka wyświetlić długość drogi z wierzchołka początkowego oraz drogę w postaci sekwencji wierzchołków,

c) dla problemu maksymalnego przepływu wyświetlić wartość uzyskanego przepływu oraz wykorzystanie przepływu w poszczególnych krawędziach.

Sprawozdanie powinno zawierać:

- krótki wstęp zawierający oszacowanie złożoności obliczeniowej poszczególnych problemów na podstawie literatury,

- plan eksperymentu, czyli założenia co do wielkości struktur, sposobu generowania ich elementów, sposobu pomiaru czasu, itp.
- opis metody generowania grafu (powinna ona zapewnić spójność oraz zmienną strukturę grafu),
- wyniki należy przedstawić w tabelach oraz w formie wykresów dla każdego problemu (oddzielnie MST i najkrótsza droga w grafie). Dla każdego problemu (MST oraz najkrótsza ścieżka) należy przygotować następujące wykresy:
    a. wykresy typ1 (osobne wykresy dla każdej reprezentacji grafu) - w formie linii (połączonych punktów), których parametrem jest gęstość grafu i typ algorytmu (Kruskal/Prim lub Dijsktra/Bellman-Ford), czyli 4 x 2= 8 linii na rysunek,
    b. wykresy typ2 (osobne wykresy dla każdej gęstości grafu) – w formie linii, których parametrem jest typ algorytmu i typ reprezentacji grafu, (czyli 4 linie na każdy rysunek),
    c. podobne zasady przyjąć dla algorytmu maksymalnego przepływu (w tym przypadku będzie tylko jeden algorytm).
- wszystkie wykresy powinny pokazywać zależność czasu wykonania algorytmu (oś Y) w funkcji liczby wierzchołków (oś X),
- nie umieszczać na jednym rysunku wyników działania algorytmów dla różnych problemów,
- wnioski dotyczące efektywności poszczególnych struktur. Wskazać przyczyny rozbieżności (jeśli występują) pomiędzy złożonościami teoretycznymi a uzyskanymi eksperymentalnie,
- załączony kod źródłowy w formie elektronicznej (cały projekt wraz z wersją skompilowaną programu) oraz sprawozdanie w formie papierowej.

Ocena projektu:

3.0 – po jednym algorytmie z każdego problemu (możliwość korzystania z biblioteki STL)

4.0 – po dwa algorytmy z każdego problemu (możliwość korzystania z biblioteki STL)

4.5 – po dwa algorytmy z każdego problemu (bez STL)

5.0 – wersja obiektowa (bez STL) + algorytm znajdowania maksymalnego przepływu Forda-Fulkersona (znajdowanie ścieżek metodą przeszukiwania grafu w głąb)

5.5 – wersja obiektowa (bez STL) + algorytm znajdowania maksymalnego przepływu Forda-Fulkersona (znajdowanie ścieżek metodą przeszukiwania grafu w głąb i wszerz)