

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 Prima oraz algorytm Kruskala,
- b. wyznaczanie najkrótszej ścieżki w grafie – algorytm Dijkstry oraz algorytm Forda-Bellmana,
- 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 incydencji),
- reprezentacja listowa (lista następników/poprzedników).

Należy przyjąć następujące założenia:

- wszystkie struktury danych powinny być alokowane dynamicznie,
- 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 (liczba krawędzi w stosunku do liczby wierzchołków). Badania należy wykonać dla 5 różnych (reprezentatywnych) liczb wierzchołków V 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 zaimplementowanych operacji i zbudowanej struktury (szerzej w na ten temat w dalszej części dokumentu),
- sposoby dokładnego pomiaru czasu w systemie Windows podano na stronie http://jaroslaw.mierzwa.staff.iiar.pwr.wroc.pl/pamsi/debug_and_time.pdf
- dopuszczalnymi językami programowania są języki kompilowane do kodu natywnego (np. C, C++), a nie interpretowane lub uruchamiane na maszynach wirtualnych (np. JAVA, .NET, Python),
- używanie okienek nie jest konieczne i nie wpływa na ocenę (wystarczy wersja konsolowa),

- nie wolno korzystać z gotowych bibliotek np. STL, Boost lub innych – wszystkie algorytmy i struktury muszą być zaimplementowane przez studenta (nie kopiować gotowych rozwiązań) – oprócz pewnych wyjątków zapisanych przy ocenach
- implementacja projektu powinna być wykonana w formie jednego programu,
- kod źródłowy powinien być komentowany.

Sprawdzenie poprawności zbudowanej struktury/operacji obejmuje:

- wczytanie struktury grafu z pliku tekstowego. Plik zawiera opis poszczególnych krawędzi według wzoru: początek krawędzi, koniec krawędzi oraz waga/przepustowość. Struktura pliku jest następująca:
 - a. w pierwszej linii zapisana jest liczba krawędzi oraz liczba wierzchołków (rozdzielone spacją),
 - b. wierzchołki numerowane są w sposób ciągły od zera,
 - c. w kolejnych liniach znajduje się opis krawędzi (każda krawędź w osobnej linii) w formie trzech liczb przedzielonych spacjami (wierzchołek początkowy, wierzchołek końcowy oraz waga/przepustowość),
 - d. dla problemu MST pojedynczą krawędź traktuje się jako nieskierowaną, natomiast dla algorytmów najkrótszej drogi i maksymalnego przepływu jako skierowaną,
- losowe wygenerowanie grafu (jako dane podaje się liczbę wierzchołków oraz gęstość w %). Graf z pliku i wygenerowany losowo zajmują tę samą zmienną (czyli ostatnia operacja generowania lub wczytywania z pliku nadpisuje poprzednią),
- możliwość wyświetlenia na ekranie wczytanego lub wygenerowanego grafu w formie reprezentacji listowej i macierzowej,
- uruchomienie algorytmu dla obu reprezentacji i wyświetlenie wyników na ekranie. Dla problemu najkrótszej drogi w grafie i maksymalnego przepływu musi być możliwość podania wierzchołka początkowego i końcowego.

Poniższe operacje należy zrealizować w formie menu dla każdej struktury i dla każdego problemu z osobna):

1. Wczytaj z pliku.
2. Wygeneruj graf losowo.
3. Wyświetl listowo i macierzowo na ekranie.
4. Algorytm 1 (np. Prima) macierzowo i listowo z wyświetleniem wyników.
5. Algorytm 2 (np. Kruskala) macierzowo i listowo z wyświetleniem wyników

Uwaga! Po uruchomieniu program powinien zapytać, który problem chcemy rozwiązywać i przejść do odpowiedniego menu.

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. Sposób powinien zapewnić spójność oraz zmienną strukturę grafu,
- wyniki - należy przedstawić w tabelach oraz w formie wykresów dla każdego problemu osobno (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 Dijkstra/Bellman) - czyli $4 \times 2 = 8$ 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. analogicznie jest dla algorytmu maksymalnego przepływu.
- wszystkie wykresy (ich osie) to czas wykonania algorytmu (oś Y) w funkcji ilości wierzchołków (oś X),
- wnioski dotyczące efektywności poszczególnych struktur. Wskazać (jeśli są) przyczyny rozbieżności 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 – pod dwa algorytmy z każdego problemu (możliwość wykorzystania z biblioteki STL)
- 4.5 – pod dwa algorytmy z każdego problemu (bez STL)
- 5.0 – wersja obiektowa (bez STL) + algorytm znajdowania maksymalnego przepływu Forda-Fulkersona (wynajdywanie ścieżek metodą przeszukiwania grafu w głąb).

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