Laboratoria Algorytmów i Struktur Danych

Projekt 3

Sortowanie topologiczne grafy

Wojciech Niedziela 160363

Jan Chojnacki 159567

Godzina: **11:45** Dzień: **Piątek** Grupa: **14**

Nazwa kierunku: **Informatyka**

POLITECHNIKA POZNAŃSKA

1 | Wstęp

Celem tego sprawozdania jest zaprezentowanie jest zaprezentowanie operacji na grafach, w szczególności algorytmów sortowania topologicznego, takich jak algorytm Kahna oraz algorytm Tarjana.

2 | Tworzenie grafu

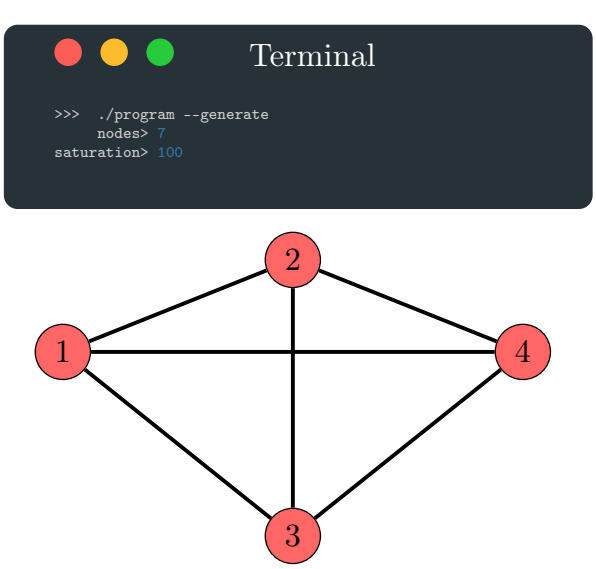
W naszym programie znajdują się dwie możliwości tworzenia grafu:

2.1 | Generacja grafu

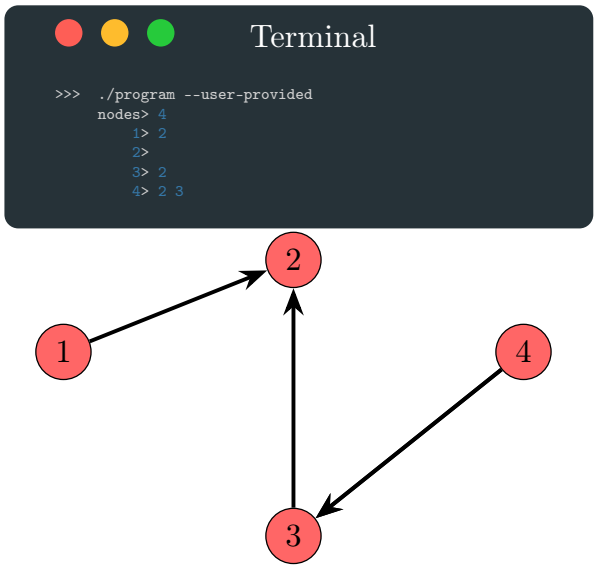
Generacja grafu tworzy graf skierowany, acykliczny, o podanej liczbie wierzchołków oraz określonym zagęszczeniu krawędzi (saturacji). Taki graf jest przechowywany w macierzy sąsiedztwa, gdzie górny trójkąt jest wypełniany odpowiednią ilością jedynek.

2.2 | Ładowanie grafu

Ładowanie grafu umożliwia użytkownikowi programu wybór formy reprezentacji grafu (lista sąsiedztwa, macierz sąsiedztwa, tabela krawędzi). Następnie podaje się liczbę wierzchołków oraz ich następników.



(a) Generowanie grafu



(b) Wczytanie grafu

3 | Operacje na grafach

Na grafach jako strukturach danych można wykonywać różne opracje takie jak:

3.1 | Wybieranie reprezentacji

W grafach używane są trzy najbardziej popularne reprezentacje przy pomocy których zapisuje się informacje o danej strukturze:

- **Lista sąsiedztwa** - Dla każdego wierzchołka w grafie przechowuje się listę wierzchołków, do których prowadzi krawędź. Jest to efektywne dla grafów rzadkich (mających mało krawędzi) i umożliwia szybkie wyszukiwanie wszystkich sąsiadów danego wierzchołka.
- **Macierz sąsiedztwa** - Jest to dwuwymiarowa tablica, w której komórka w i-tym wierszu i j-tej kolumnie wskazuje, czy w grafie istnieje krawędź między wierzchołkiem i a wierzchołkiem j. Umożliwia ona szybkie sprawdzenie, czy pomiędzy dowolnymi dwoma wierzchołkami istnieje krawędź, ale w przypadku grafów o dużej liczbie wierzchołków może zajmować dużo pamięci.
- **Tabela krawędzi** - Jest to lista wszystkich krawędzi w grafie, gdzie każda krawędź jest reprezentowana jako para wierzchołków. Jest to szczególnie użyteczne, gdy chcemy przetwarzać krawędzie grafu.

3.2 | Wypisywanie grafu

Sposób wyświetlania grafu zależy od jego rezprezentacji. Mogą być one zapisane w trzech formatach: lista sąsiedztwa, macierz sąsiedztwa oraz tabela krawędzi.

- **Lista sąsiedztwa** - Funkcja iteruje przez wszystkie klucze (wierzchołki) i drukuje je wraz z odpowiadającymi im listami sąsiednich wierzchołków. Dla każdego klucza funkcja wyświetla wierzchołek oraz jego sąsiadów w formacie: "wierzchołek: sąsiad1, sąsiad2, ...".
- **Macierz sąsiedztwa** - Funkcja drukuje dwuwymiarową tablicę reprezentującą macierz sąsiedztwa. Pierwsza linia zawiera nagłówki kolumn z numerami wierzchołków. Następnie, dla każdego wiersza, funkcja wyświetla numer wierzchołka i odpowiednie wartości z macierzy, gdzie 1 oznacza istnienie krawędzi, a 0 jej brak.
- **Tabela krawędzi** - Funkcja drukuje listę krawędzi, gdzie każda krawędź jest reprezentowana jako para wierzchołków w formacie: "wierzchołek1 -> wierzchołek2".

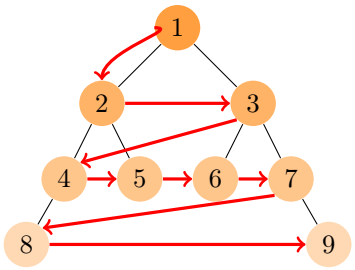
3.3 | Wyszukiwanie krawędzi

Wyszukiwanie krawędzi jak mówi nazwa, służy do sprawdzania, czy krawędź między podanymi wierzchołkami istnieje. W zależności od reprezentacji grafu, funkcja stosuje odpowiednie metody sprawdzania.

- **Lista sąsiedztwa** - Funkcja sprawdza, czy wierzchołek `to_node` znajduje się na liście sąsiedztwa wierzchołka `from_node`. Zwraca wartość `True` jeśli krawędź istnieje, w przeciwnym razie `False`.
- **Macierz sąsiedztwa** - Funkcja sprawdza wartość w macierzy sąsiedztwa w komórce znajdującej się na przecięciu wiersza odpowiadającego `from_node` i kolumny odpowiadającej `to_node`. Jeśli wartość wynosi 1, zwraca `True`, w przeciwnym razie `False`.
- **Tabela krawędzi** - Funkcja sprawdza, czy para (`from_node`, `to_node`) istnieje w zbiorze krawędzi. Zwraca `True` jeśli krawędź jest obecna, w przeciwnym razie `False`.

3.4 | Przechodzenie wszerz(BFS)

BFS (Breadth-First Search) służy do przeszukiwania grafu wszerz od wskazanego wierzchołka. Wykorzystuje ono kolejkę do śledzenia wierzchołków, które należy odwiedzić.
Funkcja pobiera wierzchołek z początku kolejki. Jeśli wierzchołek ten nie został jeszcze odwiedzony, jest oznaczany jako odwiedzony, a następnie dodawany do listy odwiedzonych. Następnie, w zależności od typu reprezentacji grafu, funkcja dodaje sąsiadów bieżącego wierzchołka do kolejki, jeśli nie zostali jeszcze odwiedzeni.
Funkcja zwraca listę, która zawiera kolejność odwiedzanych wierzchołków w trakcie przeszukiwania wszerz.

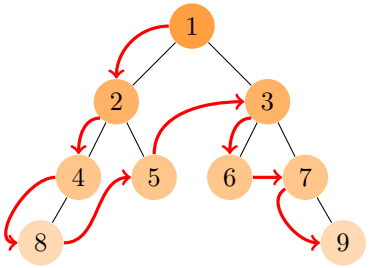


Rysunek 2: Wizualizacja przeszukiwania **w głąb**

W powyższym diagramie, wierzchołki są odwiedzane w kolejności wszerz, poczynając od wierzchołka 1, a następnie przechodząc do jego sąsiadów w kolejności numerycznej. Strzałki wskazują kolejność przeszukiwania.

3.5 | Przechodzenie w głąb(DFS)

DFS (Deep First Search), czyli Przeszukiwanie w Głąb, to jeden z podstawowych algorytmów trawersowania (przechodzenia) grafu. Ten algorytm odwiedza wierzchołki, przechodząc jak najgłębiej, zanim wróci do poprzedniego wierzchołka i przejdzie do kolejnego, jeszcze nieodwiedzonego sąsiada.



Rysunek 3: Wizualizacja przeszukiwania **w głąb**

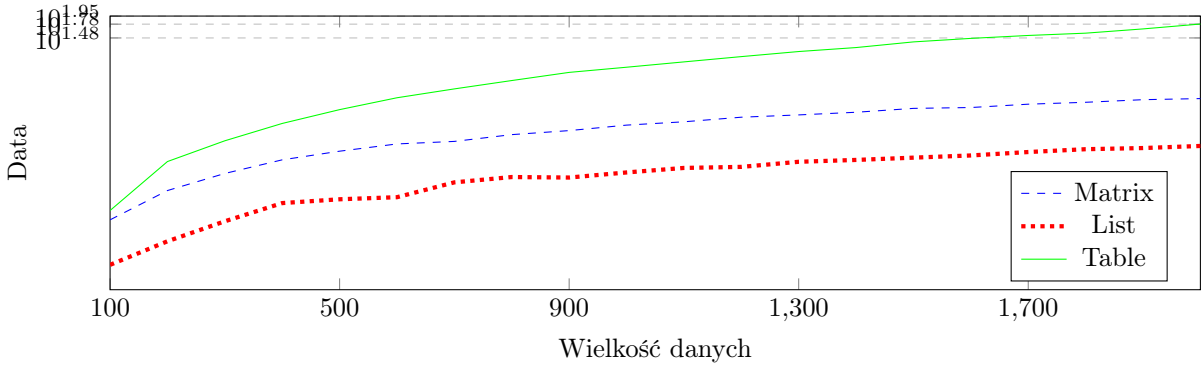
4 | Sortowania topologiczne

Sortowanie topologiczne przeprowadza się na grafie acyklicznym i skierowanym (DAG). Polega ono na liniowym uporządkowaniu wierzchołków tego grafu w taki sposób, że dla każdej pary wierzchołków (u), (v) połączonych krawędzią z (u) do (v), w porządku topologicznym (u) znajduje się przed (v).

4.1 | Algorytm Kahna

Algorytm Kahna to skuteczna metoda sortowania topologicznego, oparta na usuwaniu wierzchołków bez krawędzi wejściowych z grafu skierowanego. W implementacji funkcji khan oblicza się stopień wejściowy dla każdego wierzchołka, a następnie iteracyjnie usuwa się wierzchołki bez krawędzi wejściowych, aktualizując stopnie wejściowe sąsiadujących wierzchołków.
Jeśli wszystkie wierzchołki nie zostaną posortowane, oznacza to obecność cyklu w grafie.

Sortowanie topologiczne algorytmem Kahna



4.2 | Algorytm Tarjana

Algorytm Tarjana jest oparty na metodzie przeszukiwania drzewa DFS (Deep First Search). Algorytm zaczyna od wierzchołka niezależnego (nie mającego krawędzi wchodzących), a następnie odwiedza jego następnika. Następnie szuka następnika właśnie odwiedzonego wierzchołka. Operacja ta jest wykonywana, dopóki wierzchołek ma następnika. Jeżeli go nie ma, wierzchołek jest dodawany na stos oraz oznaczany jako wierzchołek będący już na stosie. Następnie cofamy się do poprzedniego wierzchołka i powtarzamy algorytm.

Jeżeli na danym etapie algorytmu w grafie nie ma żadnego wierzchołka niezależnego, oznacza to, że w danym grafie znajduje się cykl.

Sortowanie topologiczne algorytmem Tarjana