Helena Masłowska gr. 5.1
indeks: 148182
2 sem. Informatyka

Opracowanie trzech reprezentacji maszynowych grafu

Repozytorium:
https://github.com/HelenaMaslowska/Reprezentacje_maszynowe_grafu

Spis treści

Wstęp

Celem niniejszego opracowania jest porównanie i ocena efektywności poszczególnych reprezentacji maszynowych grafu. Opracowane zostały 3 reprezentacje: macierz sąsiedztwa, lista krawędzi oraz lista sąsiedztwa. Dane zostały wygenerowane przez program sprawdzający cykliczność tworzonego grafu z każdą dodaną krawędzią. Wygenerowane dane to lista par wierzchołków, które reprezentują kolejne łuki grafu. Każdy taki graf zawiera 50% możliwych do utworzenia łuków. Rzędy grafów poddanych testowi były z zakresu 100 - 1000.

Sposób generowania danych wejściowych

Aby opracować wymienione wyżej reprezentacje, należy dobrać odpowiednie dane wejściowe. W tym celu potrzebny jest program do generowania danych wejściowych, który w trakcie dodawania nowego łuku sprawdzi, czy dodany łuk nie naruszy acykliczności grafu.

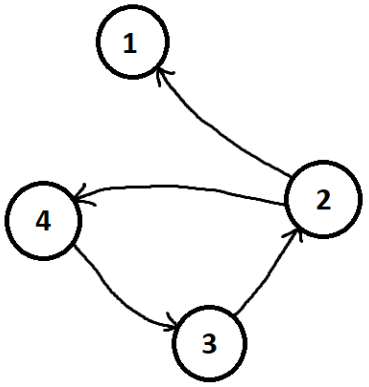
Metodę, która została użyta do generacji danych opisuje się następująco: należy stworzyć pusty graf, gdzie będą dodawane łuki. Wygenerowanie grafu acyklicznego to wylosowanie dowolnej pary wierzchołków (która ma reprezentować łuk w nowym grafie) i sprawdzenie czy gdy zostanie on dodany do grafu acykliczność grafu zostanie zachowana. Jeśli łuk burzy konwencję acykliczności to program ponownie losuje parę wierzchołków. Proces ten jest wykonywany aż nie uzyska się wymaganej ilości łuków w grafie. Żeby program nie działał w nieskończoność bądź zapętlił, została narzucona z góry liczba losowań, która wynosi 1 milion. Metoda, która została użyta do przeszukiwania czy graf spełnia warunki zadania to DFS.

Mały opis algorytmu, gdzie stosuje się DFS: początkowo każdy z wierzchołków w grafie jest “kolorowany” na biało, czyli jako niesprawdzony. Sprawdzenie polega na zamalowaniu aktualnego wierzchołka na szaro jeśli wcześniej był biały oraz sprawdzeniu kolejnych wierzchołków. W przypadku gdy program natrafi na wierzchołek już pomalowany na szaro zwraca fałsz, ponieważ znalazł cykl.

W implementacji algorytmu graf jest reprezentowany jako lista sąsiedztwa.

Macierz sąsiedztwa

		wierzch. końcowy			
		1	**2**	**3**	**4**
wierzch. startowy	**1**	0	0	0	0
	2	1	0	0	1
	3	0	1	0	0
	4	0	0	1	0



Dane na temat łuków / krawędzi w grafie są zapisane w formie tabeli (przykład powyżej) gdzie wiersze odpowiadają wierzchołkom startowym, a kolumny wierzchołkom końcowym w grafie.

Złożoności operacji na macierzy sąsiedztwa

- złożoność pamięciowa: $O(V^2)$
- iteracja po wszystkich łukach: $O(V^2)$
- przejrzenie następników danego wierzchołka: $O(V)$
- sprawdzenie istnienia jednego łuku: $O(1)$

Średni/pesymistyczny przypadek

Pesymistyczny przypadek w tej reprezentacji to duża liczba wierzchołków w grafie, gdyż będzie trzeba zużyć ogromną ilość pamięci w komputerze. Złożoność pamięciowa wynosi V^2, a zatem im więcej wierzchołków tym więcej miejsca będzie potrzebować. Reprezentacja ta jest najlepsza dla grafów o małym rzędzie.

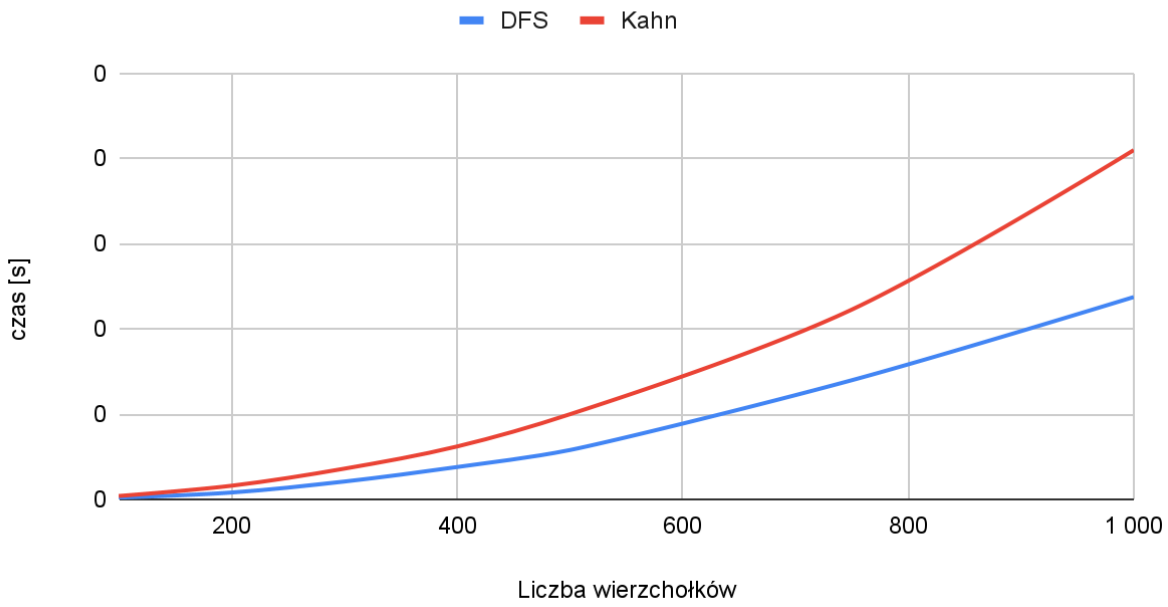
Zalety

Zdecydowanie największą zaletą takiego rozwiązania jest czytelność zapisanych danych, prostota w implementacji oraz możliwość zastosowania rozwiązania dla grafu nieskierowanego, skierowanego i ważonego. Bardzo łatwo można też znaleźć informację o wierzchołku, a złożoność w przypadku sprawdzenia czy istnieje łuk wynosi O(1), gdyż wszystkie informacje o grafie można sprawdzić przeszukując odpowiednią kolumnę i wiersz. Najlepszy dla grafu gęstego.

Wady

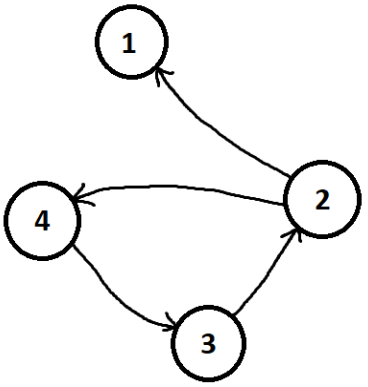
Przy większych danych wejściowych czy dużym rzędzie grafu zajmuje dużo pamięci w komputerze.

Macierz sąsiedztwa - wykres liniowy



Lista krawędzi

```
[ [2,1], [3,2], [4,3], [2,4] ]
```



Dane są zapisywane w formie tablicy, gdzie każdy element jest parą (startowy wierzchołek, końcowy wierzchołek) przedstawiającą łuk w grafie.

Złożoności operacji na listach krawędzi

- złożoność pamięciowa: O(E)
- iteracja po wszystkich łukach: O(E)
- przejrzenie wszystkich następników danego wierzchołka: O(E)
- sprawdzenie istnienia jednego łuku: O(E)

Średni/pesymistyczny przypadek

Pesymistyczny przypadek w tej reprezentacji to duża liczba łuków w grafie. Dlatego, najlepiej jest korzystać z tej reprezentacji, gdy mamy do czynienia z grafem rzadkim. Reprezentacja wymaga pewnych udoskonaleń jak np. sortowanie łuków po startowym łuku, aby zmniejszyć liczbę obliczeń w konkretnych operacjach np. złożoność obliczeniową istnienia jednego łuku.

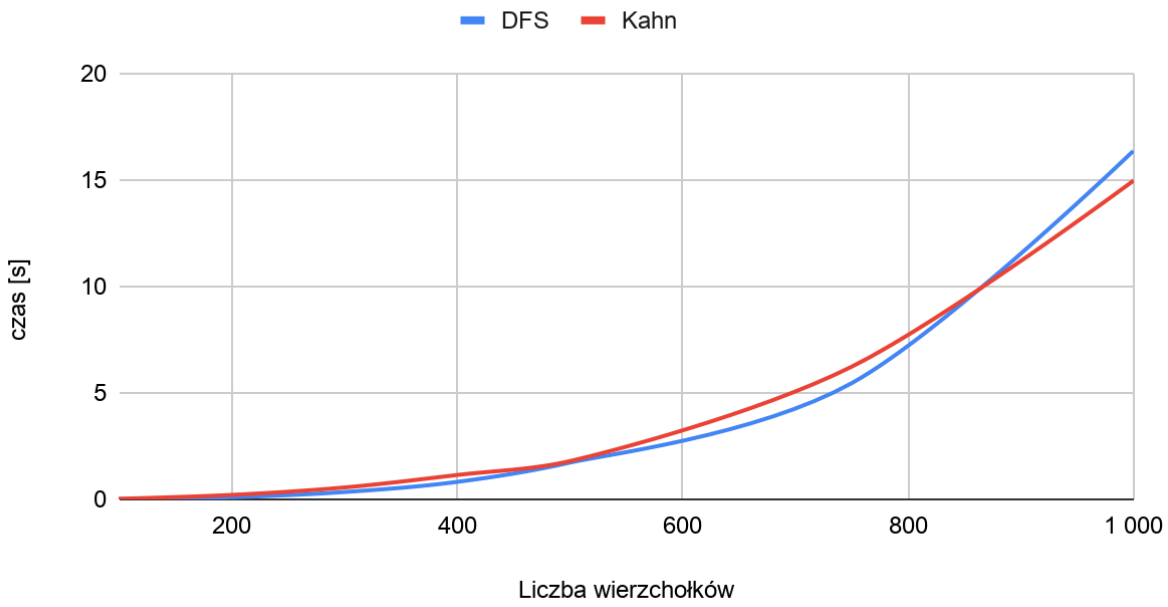
Zalety

Największą zaletą takiego rozwiązania jest prostota w implementacji. Porządkowanie wprowadzonych danych wymaga jedynie implementacji algorytmu sortowania.

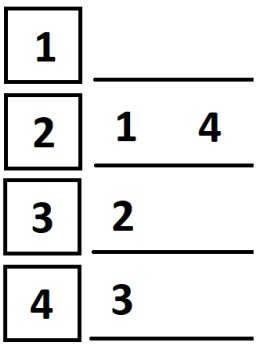
Wady

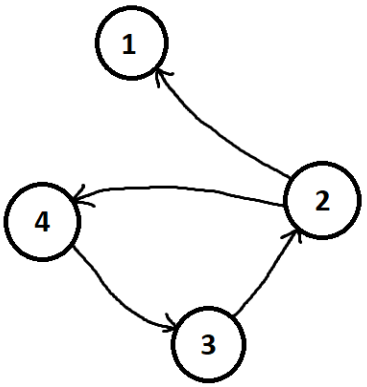
Lista krawędzi to zbiór par wierzchołków przez co trudno poruszać się po takiej reprezentacji. Trudno również sprawdzić czy istnieje dany łuk bez sprawdzenia wszystkich krawędzi. Iteracja po łukach jest uciążliwa, tak samo jak przejrzenie wszystkich następników danego węzła. W tym celu potrzebna jest tablica pomocnicza, która “gromadzi” znalezione następniki wierzchołka.

Lista krawędzi - wykres liniowy



Lista sąsiedztwa





Reprezentacja grafu przez listy sąsiedztwa polega na zapisaniu następnika wierzchołka na liście danego poprzednika (jak narysowano powyżej). Lista w kwadracie to indeksy tablicy list, w których zapisujemy kolejne następniki danego wierzchołka.

Złożoności operacji na listach sąsiedztwa

- złożoność pamięciowa: O(V + E)
- iteracja po wszystkich łukach: O(E)
- przejrzenie następników jednego wierzchołka: pesymistycznie O(V) (tyle sąsiadów może mieć wierzchołek), średnio O(E/V) (trzeba przejrzeć całą listę sąsiadów, których średnio wierzchołek ma E/V)
- sprawdzenie istnienia jednego łuku: pesymistycznie O(V) (jak powyżej, gdyż wszystkie wierzchołki mogą być połączone łukiem z jednym poprzednikiem, a sprawdzany wierzchołek końcowy może być pesymistycznie na końcu listy połączonych łuków), średnio O(E/V)

Średni/pesymistyczny przypadek

Pesymistyczny przypadek w tej reprezentacji to graf wzbogacony o dużą liczbę łuków, przez co iteracja po łukach może się wydłużyć, złożoności zostały omówione powyżej.

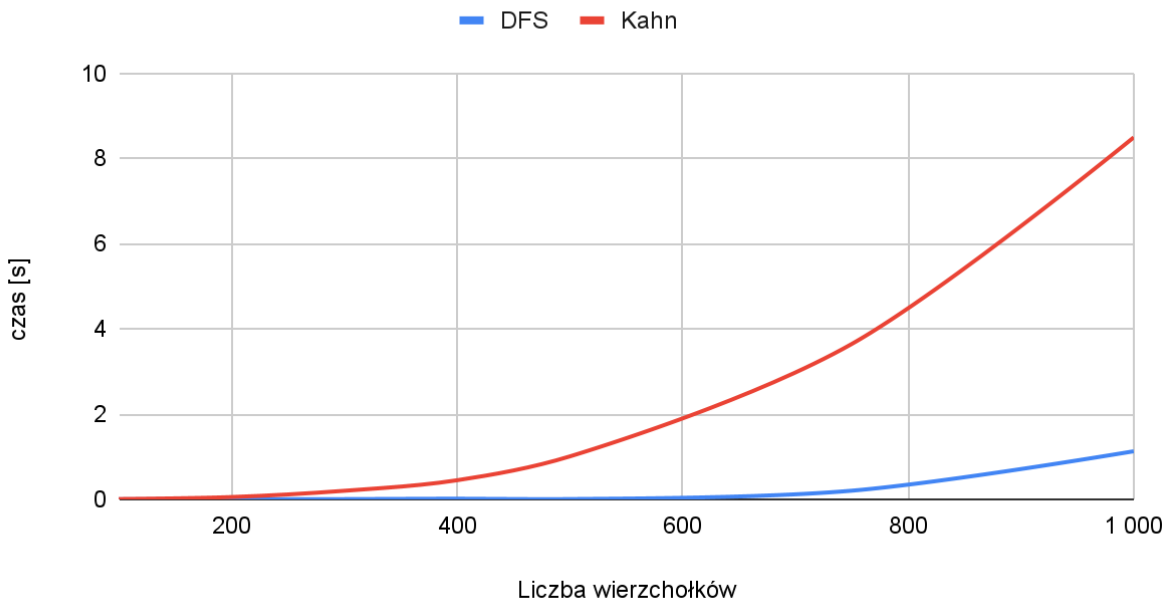
Zalety

Lista sąsiedztwa zajmuje mniej pamięci komputera niż macierz sąsiedztwa, ponadto jest bardziej czytelna niż lista krawędzi. Umożliwia szybki dostęp do następników danego wierzchołka.

Wady

Długi czas sprawdzenia istnienia pojedynczego łuku, co można poprawić do złożoności O(log V) wprowadzając drzewa BST lub AVL.

Lista sąsiedztwa - wykres liniowy



Wnioski

Każda reprezentacja grafu może być przydatna w zależności od tego, jakie mamy na dane wejściowe i co chcemy z nimi dalej zrobić. Macierz sąsiedztwa zajmuje dużo pamięci ale działa dość szybko, więc nadaje się do grafów o małym rzędzie, bądź grafów gęstych. Listę krawędzi można użyć, gdy chcemy szybko posortować łuki w grafie. Listy sąsiedztwa - jeśli chcemy szybko przeszukać łuki albo wierzchołki w grafie.