

Kierunek: **Informatyka algorytmiczna (INA)**

Specjalność:

PRACA DYPLOMOWA
INŻYNIERSKA

**Zastosowanie szkiców danych w
analizie dużych grafów**

**Application of data sketches in the
analysis of large graphs**

Paweł Polerowicz

Opiekun pracy
dr inż. Jakub Lemiesz

Słowa kluczowe: TODO, TODO, TODO

Streszczenie

Polski

Słowa kluczowe: TODO, TODO, TODO

Abstract

English

Keywords: TODO, TODO, TODO

Spis treści

1. Wstęp	8
1.1. Struktura pracy	8
2. Opis problemu	9
2.1. Analiza grafów	9
2.2. Główne sposoby modelowania problemu	9
2.3. Definicja formalna	9
3. Przegląd literatury	10
3.1. Rozmiar i układ treści na stronach dokumentu	10
3.2. MDL	10
3.3. Metody oparte na modyfikacji macierzy sąsiedztwa	10
3.4. Embeddings	11
4. Szkice danych	12
4.1. Definicja	12
5. Główny przedmiot pracy	13
5.1. Co nas najbardziej interesuje	13
5.2. Jaki jest ogólny pomysł	13
5.3. Jak to dokładnie zrobiliśmy	13
6. Analiza wyników	14
6.1. Architektura eksperymentów	14
6.2. Wyniki	14
6.3. Wnioski	14
Literatura	15

Spis rysunków

Spis tabel

Spis listingów

Skróty

ISO (ang. *International Standards Organization*)

Rozdział 1

Wstęp

TODO: Tutaj będzie bardzo ogólne wprowadzenie do pracy.

- Motywacja
- Krótki i "wysokopoziomowy" opis problemu
- Podsumowanie osiągnięć pracy

1.1. Struktura pracy

TODO: Kilka(dziesiąt) słów o strukturze i zawartości pracy. Omówienie po kolei rozdziałów i ewentualnych dodatków. Bardzo skrótowo, bo wszystko będziemy potem i tak rozwijać. Coś w jak niżej (do dopracowania).

Pierwszy rozdział stanowi niniejszy wstęp, przedstawiający ogólny zarys problematyki pracy i skrótowo podsumowujący jej wkład badawczy. W drugim rozdziale znajduje się opis problemu wraz z formalną definicją i przedstawieniem różnych jego wariantów. Przedmiotem trzeciego rozdziału jest przegląd literatury związanej z analizą wielkich grafów, z podziałem na zastosowane metodyki oraz tabelą ilustrującą porównanie znanych struktur i algorytmów. W czwartym rozdziale szczegółowo omówione zostały szkice danych, od ich formalnej definicji do bardziej praktycznych przykładów ich wykorzystanie, także w kontekście niniejszej pracy. Piąty rozdział zawiera właściwy opis tego, co zostało zrobione [TODO: przepisać nieco bardziej szczegółowo]. W rozdziale szóstym opisane zostały przeprowadzone eksperymenty, wraz z prezentacją wyników oraz wnioskami z nich płynącymi. Ostatni, siódmy rozdział, stanowi podsumowanie pracy. Zawarte zostały w nim ogólne konkluzje na temat pracy oraz możliwe kierunki dalszych badań. Pracy towarzyszy wykaz literatury oraz dodatek, zawierający opis dołączonej płyty CD [TODO: czy nadal załączamy płytę?] i instrukcję użytkowania części implementacyjnej.

Rozdział 2

Opis problemu

2.1. Analiza grafów

2.2. Główne sposoby modelowania problemu

2.3. Definicja formalna

Rozdział 3

Przegląd literatury

3.1. Rozmiar i układ treści na stronach dokumentu

3.2. MDL

Sketch synopses - CountMin, gSketch Graph summaries - Graph pattern matching over streams
Graph stream algorithms Distributed graph systems

3.3. Metody oparte na modyfikacji macierzy sąsiedztwa

Jednym z najbardziej popularnych i być może najprostszym koncepcyjnie sposobem reprezentacji grafu jest macierz sąsiedztwa. Jej wiersze i kolumny odpowiadają poszczególnym wierzchołkom, a w komórkach przechowywane są wagi krawędzi pomiędzy nimi, o ile takowe krawędzie istnieją. W przypadku grafów nieważonych może to być np. wartość logiczna indykująca istnienie krawędzi lub ustalona stała. Ten sposób reprezentacji ma niewątpliwe zalety takie jak prostota implementacji i, przede wszystkim, stały czas dostępu do wag krawędzi. Z tego powodu

Jedną z pierwszych realizacji tej idei o przechowywania informacji o strumieniowanym grafie jest struktura TCM[2]. Ma ona postać macierzy o boku długości m , gdzie m jest pewną stałą. Podobnie jak w klasycznej macierzy sąsiedztwa, w jej komórkach składowane są wagi krawędzi. Zasadniczą różnicą jest natomiast sposób wyznaczania rzędu i kolumny odpowiadających danej parze wierzchołków. Są one bowiem wyznaczane przez wynik funkcji haszującej $H : V \rightarrow [1..m]$. Czas obliczania hasza jest stały, a co za tym idzie, złożoność czasowa zapytań i dodawania nowych krawędzi również. Teoretyczna złożoność pamięciowa także jest stała i wynosi $O(m^2)$. W praktycznych zastosowaniach wybór m zależy jednak często od liczby krawędzi i przejmuje się najczęściej m rzędu $O(\sqrt{|V|})$. Dokładność rezultatów zależy od rozmiaru macierzy i może być niska ze względu na kolizje haszy. Łatwo zauważyć, że jeśli m jest istotnie mniejsze od $|V|$ to może do nich dochodzić często, co powoduje traktowanie różnych krawędzi jako kolejnych instancji tego samego połączenia. Autorzy, świadomi tego ograniczenia, proponują zastosowanie kilku parami niezależnych funkcji haszujących i stworzenie na ich podstawie wielu szkiców grafu. Przykładowo, jeśli badaną zmienną jest suma wag kolejnych instancji krawędzi między danymi dwoma wierzchołkami, to algorytm może sprawdzić odpowiednie komórki dla wszystkich szkiców, a następnie zwrócić maksymalną wartość. Podejście to pozwala na analizę większych grafów niż w przypadku pojedynczego szkicu, ale ostatecznie nie rozwiązuje całkowicie problemu. Użyteczność struktury TCM w bazowej formie jest dyskusyjna, stanowi ona jednak punkt wyjściowy dla bardziej zaawansowanych rozwiązań.

Strukturą opartą na koncepcie podobnym do TCM jest *Graph Stream Sketch* (GSS)[1]. Celem autorów było stworzenie metody oferującej lepszą skalowalność dla wielkich grafów. Podobnie jak w TCM, funkcja haszująca mapuje zbiór wierzchołków na pewien mniejszy zbiór M -elementowy. Rozmiar macierzy jest natomiast równy m , $m < M$. Główną zmianą jest wprowadzenie dodatkowych cech opisujących wierzchołki. Na podstawie hasza $H(v)$ wyznaczany jest podpis wierzchołka $f(v)$ ($0 \leq f(v) < F$), gdzie $M = m \times F$ i $f(v) = H(v) \% F$, a także adres $h(v) = \lfloor \frac{H(v)}{F} \rfloor$. Adresy służą do wyznaczania rzędu i kolumny komórek. Komórki te mają one postać krotki lub, bardziej obrazowo, kubelka, w którym przechowywana jest para podpisów wierzchołków tworzących krawędź oraz kumulatywna waga krawędzi. Przechowywanie podpisów w komórkach pozwala zredukować ryzyko konfliktu haszy. Łatwo bowiem zauważyć, że nawet jeśli dwa różne wierzchołki mają taki sam adres, to istnieje duża szansa, że ich podpisy są różne. Z tego względu nowa krawędź jest dodawana do kubelka tylko w wypadku, gdy jest on pusty lub gdy istniejące w nim podpisy są zgodne z podpisami wierzchołków krawędź tą tworzących. W przeciwnym przypadku, jest ona zapisywana w dodatkowym buforze, mającym postać listy sąsiedztwa pełnych haszy. Pozwala on na dodawanie nowych krawędzi z niskim ryzykiem kolizji, nawet jeśli sama macierz jest już wypełniona. Należy natomiast zauważyć, że część macierzowa struktury jest bardziej efektywna czasowo, oferując stały czas odpowiedzi na zapytanie, podczas gdy dla bufora jest on liniowy względem liczby wierzchołków. Dokładność odpowiedzi w części macierzowej zależy od długości podpisów. Potencjalnym problemem GSS jest niskie wykorzystanie pamięci w macierzy. Przy kolizji adresów nowe krawędzie mogą trafiać do bufora, mimo, że w samej macierzy pozostaje wiele pustych komórek. Aby temu zaradzić, autorzy proponują haszowanie krzyżowe (*ang. square-hashing*). Zakłada ona obliczanie dla każdego wierzchołka sekwencji niezależnych adresów. Podczas wstawiania nowych krawędzi algorytm sprawdza nie jedną komórkę macierzy, a kilka, zgodnie z sekwencją adresów i wybiera pierwszą spełniającą wymagania co do zgodności podpisów.

3.4. Embeddings

Rozdział 4

Szkice danych

4.1. Definicja

Rozdział 5

Główny przedmiot pracy

5.1. Co nas najbardziej interesuje

5.2. Jaki jest ogólny pomysł

5.3. Jak to dokładnie zrobiliśmy

Rozdział 6

Analiza wyników

6.1. Architektura eksperymentów

6.2. Wyniki

6.3. Wnioski

Literatura

- [1] X. Gou, L. Zou, C. Zhao, T. Yang. Fast and accurate graph stream summarization. *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, 2019.
- [2] N. Tang, Q. Chen, P. Mitra. Graph stream summarization. *Proceedings of the 2016 International Conference on Management of Data*, 2016.