Zastosowanie szkiców danych w analizie dużych grafów

Paweł Polerowicz

Praca napisana pod kierunkiem **dr. inż. Jakuba Lemiesza**

3 lipca 2024

Główne sposoby modelowania problemu

- Model klasyczny
- Model strumieniowy
- Model półstrumieniowy
- Model rozproszony

Definicja 1 (Strumień grafowy)

Strumieniem grafowym nazywamy ciągłą sekwencję elementów, z których każdy ma postać trójki:

$$e_i = (< s_i, d_i >; w_i, t_i),$$

gdzie s_i, d_i wierzchołki grafu i przez parę $< s_i, d_i >$ oznaczamy krawędź pomiędzy nimi. Z kolei w_i i t_i to odpowiednio waga tej krawędzi i moment jej wystąpienia.

Cele pracy

- Przegląd istniejących rozwiązań w dziedzinie analizy strumieni grafowych.
- Generowanie zanurzeń wierzchołków.
- Użycie schematu szkicowania opartego na próbkach z rozkładu wykładniczego.
- Wykorzystanie operacji teoriomnogościowych do efektywnego operowania na szkicach danych.
- Analiza precyzji rekonstrukcji krawędzi grafu oraz złożoności obliczeniowej.

Zanurzenia wierzchołków

Definicja 2

Zanurzeniem wierzchołka v nazywamy wektor $\overline{S}^v \in \mathbb{R}_+^m$*, wyznaczany na podstawie cech danego wierzchołka. Będziemy przyjmować* $m \ll |V|$.

- Zanurzenia wierzchołków stanowią efektywną pamięciowo reprezentację grafu – złożoność $O(m|V|)$.
- Mogą być tworzone na różne sposoby, np. próbkowanie wierzchołków i spacery losowe, faktoryzacja macierzy sąsiedztwa, **wykorzystanie funkcji haszujących do generowania próbek z rozkładu wykładniczego**.
- Taka reprezentacja może zostać łatwo wykorzystana jako dane w uczeniu maszynowym.

NodeSketch (Yang, Rosso, Li, Cudre-Mauroux, 2019)

Algorithm NodeSketch($\tilde{A}, k, \alpha$)

if $k > 2$ **then**
- $S(k-1) \leftarrow \mathit{NodeSketch}(\tilde{A}, k-1, \alpha)$ **foreach** *rząd* r *w* $\tilde{A}$ **do**
  - $\tilde{V}_i^r(k) \leftarrow \tilde{V}_i^r + \sum_{n \in \Gamma(r)} \frac{\alpha}{m} \sum_{j=1}^{m} \mathbb{1}_{[S_j^n(k-1)=i]}, i \in \{1, 2, \ldots, D\}$
  - $S_j^r(k) \leftarrow \arg\min_{i \in \{1,2,\ldots,D\}} \frac{-\log h_j(i)}{\tilde{V}_i^r(k)}, j \in \{1, 2, \ldots, m\}$

else if $k = 2$ **then**
- **foreach** *rząd* r *w* $\tilde{A}$ **do**
  - $S_j^r(2) \leftarrow \arg\min_{i \in \{1,2,\ldots,D\}} \frac{-\log h_j(i)}{\tilde{V}_i^r(2)}, j \in \{1, 2, \ldots, m\}$

return $S(k)$

ExpSketch (Lemiesz, 2021)

Metoda generowania próbek dla krawędzi i o wadze λ_i w algorytmie `ExpSketch`:

$$E = -\frac{\ln(h(i||k))}{\lambda_i} \sim Exp(\lambda_i).$$

Twierdzenie 1 (Suma szkiców)

Niech A, B - szkice zbiorów $\mathbb{A}, \mathbb{B}$. Wtedy szkic ich sumy możemy wyznaczyć (z własności rozkładu wykładniczego) jako:

$$A \cup B = (\min\{A_1, B_1\}, \min\{A_2, B_2\}, \ldots, \min\{A_m, B_m\}).$$

ExpSketch (Lemiesz, 2021)

Definicja 3 (Ważone podobieństwo Jaccarda)

Ważone podobieństwo Jaccarda to miara podobieństwa dwóch zbiorów, zdefiniowane jako stosunek sumy wag elementów wspólnych do sumy wag elementów w sumie mnogościowej zbiorów.

$$J_w(\mathbb{A}, \mathbb{B}) = \frac{|\mathbb{A} \cap \mathbb{B}|_w}{|\mathbb{A} \cup \mathbb{B}|_w}.$$

Twierdzenie 2

Nieobciążony estymator ważonego podobieństwa Jaccarda zbiorów $\mathbb{A}$ i $\mathbb{B}$ można otrzymać, wykorzystując ich szkice:

$$\hat{J}_w(A, B) = \frac{1}{m} \sum_{k=1}^{m} \mathbb{1}[A_k = B_k].$$

EdgeSketch

Algorithm EdgeSketch($\tilde{A}, m$)

```
foreach rząd r w Ã do
    ns ← [] ;                                   // lista sąsiadów
    foreach i ∈ {1, ..., |V|} do
        if Ã[r, i] ≠ 0 then
            ns ← ns ∪ {((min(i, r)||max(i, r)), Ã[r, i])}
    S^r ← FastExpSketch(ns, m)
return S
```

Lemat (Złożoność czasowa)

Złożoność czasowa `EdgeSketch` *wynosi ogółem* $O(m(|V|)^2)$, *a w średnim przypadku dla grafów nieważonych:*

$$O((|V|)^2 + |V|(m\ln(m)\ln(|V|)))$$

EdgeSketch

- Miarą skuteczności algorytmów była precyzja rekonstrukcji krawędzi grafu.
- Rekonstrukcja wierzchołków – oblicznie macierzy podobieństw Jaccarda zbiorów reprezentujących k-sąsiedztwa wierzchołków i wybór t najwyższych wartości.
- Ostateczna macierz podobieństw w algorytmie `EdgeSketch` powstaje na podstawie macierzy niższych rzędów:

$$simM = \sum_{k=2}^{K} \alpha^{k-2} simM_k.$$

- W algorytmie `NodeSketch` jest ona obliczana raz, na podstawie zanurzeń wygenerowanych dla danego k.

Wpływ rozmiaru szkicu na precyzję rekonstrukcji

		NodeSketch			EdgeSketch		
b	**k**	**t = 1000**	**t = 10000**	**t = |E|**	**t = 1000**	**t = 10000**	**t = |E|**
4	2	0.542	0.5343	0.5131	1	0.3352	0.1547
	3	0.483	0.4991	0.5003	1	0.5342	0.4087
	4	0.499	0.4983	0.5008	1	0.5342	0.4087

Tabela: Precyzja uzyskiwana przez algorytmy NodeSketch i EdgeSketch dla grafu w stochastycznym modelu blokowym w zależności od wielkości próbek t. Parametry grafu:

- 1000 wierzchołków podzielonych na $b = 4$ bloki.
- Prawdopodobieństwo krawędzi wewnątrz bloku $p = 0.5$ i pomiędzy blokami $q = 0.001$.
- Średni stopień wierzchołka ok. 125.
- Rozmiar szkicu $m = 10$.

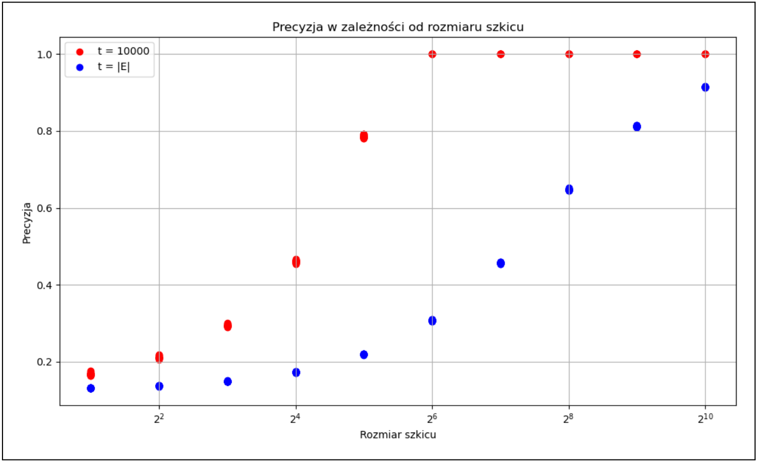
Wpływ rozmiaru szkicu na precyzję rekonstrukcji

		NodeSketch			EdgeSketch		
p	**k**	**t = 1000**	**t = 10000**	**t = |E|**	**t = 1000**	**t = 10000**	**t = |E|**
0.001	2	0	0.0016	0.0055	1	1	0.3721
	3	0.017	0.012	0.006	1	0.9671	0.4235
	4	0.003	0.0084	0.0099	1	0.9714	0.4275

Tabela: Precyzja uzyskiwana przez algorytmy NodeSketch i EdgeSketch dla grafu ważonego w modelu Erdosa-Renyiego. Parametry grafu:

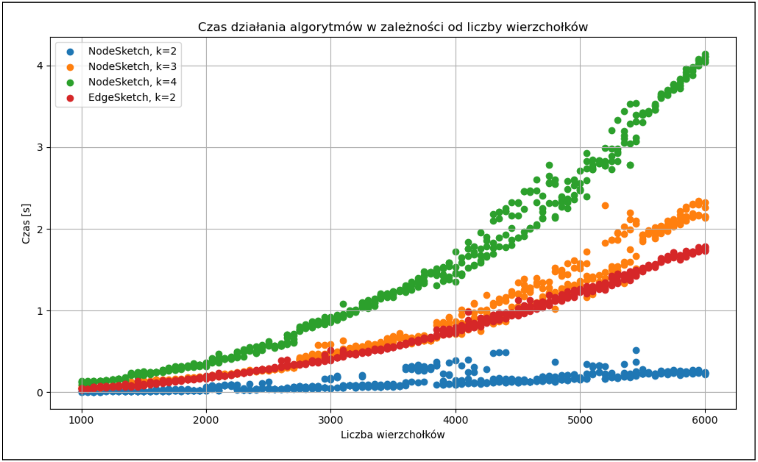
- 10000 wierzchołków.
- Prawdopodobieństwo krawędzi $p = 0.001$.
- Średni stopień wierzchołka ok. 10.
- Rozmiar szkicu $m = 10$.

Wpływ rozmiaru szkicu na precyzję rekonstrukcji



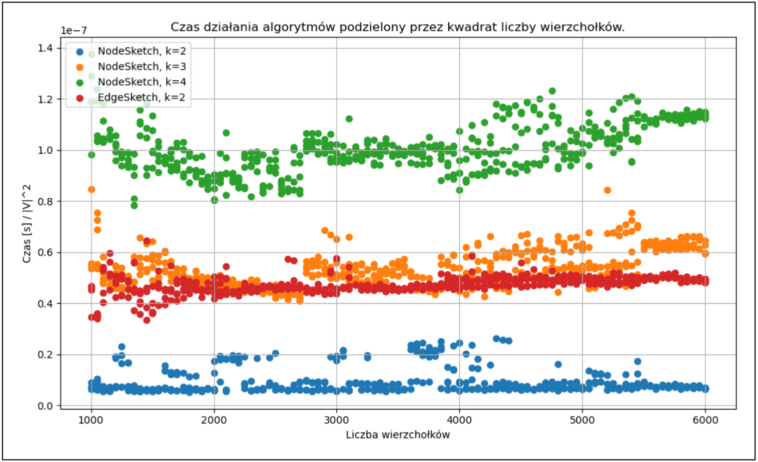
Rysunek: Precyzja uzyskiwana przez algorytm EdgeSketch w zależności od rozmiaru szkicu dla grafu w stochastycznym modelu blokowym.

Złożoność obliczeniowa



Rysunek: Czas działania algorytmów w zależności od liczby wierzchołków.

Złożoność obliczeniowa



Rysunek: Czas działania algorytmów podzielony przez kwadrat liczby wierzchołków.

Dziękuję za uwagę.