

Query-by-Sketch: Scaling Shortest Path Graph Queries on Very Large Networks

Paweł Polerowicz

Styczeń 2024

1 Wiadomości wstępne

1.1 Informacje o artykule

- Tytuł: Query-by-Sketch: Scaling Shortest Path Graph Queries on Very Large Networks
- Autorzy: Ye Wang, Qing Wang, Henning Koehler, Yu Lin
- Data publikacji: 2021
- miejsce publikacji: Proceedings of the 2021 International Conference on Management of Data

1.2 Słowniczek pojęć

Oznaczenie	Znaczenie
$G(V, E)$	graf (tu nieskierowany i spójny)
P_{uv}	zbiór najkrótszych ścieżek między wierzchołkami u i v
$d_G(u, v)$	długość najkrótszej ścieżki między u i v
$R \subset V$	zbiór punktów orientacyjnych (ang. <i>Landmarks</i>)
$L(v) = \{(r_1, \delta_{vr_1}), \dots, (r_n, \delta_{vr_n})\}$	zbiór etykiet wierzchołka v
δ_{vr_i}	$= d_G(v, r_i)$
$L = \{L(v)\}_{v \in V}$	etykietowanie nad G
$size(L) = \sum_{v \in V} L(v) $	rozmiar etykietowania

1.3 Plan prezentacji

W ramach niniejszej prezentacji zreferujemy artykuł *Query-by-Sketch: Scaling Shortest Path Graph Queries on Very Large Networks*, przedstawiający metodę QbS. Składa się ona z trzech algorytmów. Pierwszy z nich służy jako przetwarzanie wstępne i są wykonywany raz i wyznacza etykietowanie L dla grafu G . Drugi tworzy szkic danych dla pary wierzchołków u i v na podstawie etykietowania L . Ostatni wyznacza dokładną odpowiedź, wykorzystując przeszukiwanie kierowane szkicem. Złożoności algorytmów to kolejno $O(|R||E|)$, $O(|R|^4)$ (z możliwością ograniczenia do $O(|R|^2)$) oraz $O(|E| + |R||V|)$, gdzie R jest zbiorem punktów orientacyjnych.

1.4 Motywacja

Podczas poprzednich prezentacji omawialiśmy struktury wykorzystujące szkice danych do efektywnego pamięciowo przechowywania informacji o strumieniowanym grafie. W szczególności rozważaliśmy takie konstrukcje, które pozwalały na uzyskiwanie szybkich odpowiedzi na zapytania o istnienie i wagę krawędzi między dwoma wierzchołkami,

a także łączną wagę krawędzi wchodzących i wychodzących z danego wierzchołka. Niemniej jednak często zależy nam na wykonywaniu bardziej skomplikowanych operacji. Jedną z typowych może być wyznaczanie najkrótszych ścieżek między wierzchołkami. Rozwiązania tego problemu mogą znaleźć swoje zastosowanie np. w nawigacji GPS lub w analizie sieci społecznościowych. W obu przypadkach mamy do czynienia z grafami o nierzadko ogromnych rozmiarach. Sprawia to, że klasyczne i dobrze przebadane algorytmy mogą okazać się nieskuteczne, ze względu na konieczność przeglądu zatrważającej liczby krawędzi lub wykorzystywania dodatkowej pamięci o niepraktycznym rozmiarze. Dodatkowo niejednokrotnie możemy być zainteresowani wyborem nie jednej ścieżki, lecz pewnego zbioru najkrótszych lub prawie najkrótszych ścieżek. Śledząc wysiłki autorów artykułu, spróbujemy znaleźć metodę wyznaczania grafu najkrótszych ścieżek w sposób wydajny czasowo i przy użyciu rozsądnej ilości pamięci.

2 Problem

2.1 Sformułowanie problemu

2-hop distance cover (dwu-przeskokowe pokrycie odległości :))

Etykietowanie L grafu G stanowi 2-hop distance cover, jeśli zachodzi

$$\forall_{u,v \in V} d_G(u,v) = \min\{\delta_{ur} + \delta_{vr} : (r, \delta_{ur}) \in L(u), \delta_{vr} \in L(v)\}$$

Innymi słowy, wymagamy, aby dla każdej pary wierzchołków, ich etykiety zawierały co najmniej jeden wspólny punkt orientacyjny leżący na jednej z najkrótszych ścieżek je łączących.

SPG (graf najkrótszych ścieżek)

Dla dowolnych dwóch wierzchołków u i v , graf najkrótszych ścieżek (SPG) między u i v to podgraf G_{uv} grafu, gdzie:

1. $V(G_{uv}) = \bigcup_{p \in P_{uv}} V(p)$
2. $E(G_{uv}) = \bigcup_{p \in P_{uv}} E(p)$

Graf ten nie jest więc po prostu podgrafem indukowanym przez $\bigcup_{p \in P_{uv}} V(p)$, gdzie P_{uv} to zbiór najkrótszych ścieżek między wierzchołkami u i v . Każda jego krawędź musi być bowiem częścią najkrótszej ścieżki między u i v .

Problem SPG

Niech $G = (V, E)$ oraz $u, v \in V$. Problem SPG polega na znalezieniu odpowiedzi na zapytanie $SPG(u, v)$, czyli grafu najkrótszych ścieżek G_{uv} dla G . W niniejszej prezentacji będziemy dla uproszczenia zakładali, że graf jest nieskierowany, spójny i nie jest ważony. W ogólności jednak te ograniczenia są możliwe do zrelaksowania.

Algorithm 2: Constructing a labelling scheme \mathcal{L}

Input: $G = (V, E)$; a set of landmarks $R \subseteq V$

Output: A labelling scheme $\mathcal{L} = (M, L)$ with

$M = (R, E_R, \sigma)$.

```
1  $E_R \leftarrow \emptyset; L(v) \leftarrow \emptyset$  for all  $v \in V$ 
2 for all  $r_i \in R$  do
3    $Q_L \leftarrow \emptyset; Q_N \leftarrow \emptyset;$ 
4    $Q_L.\text{push}(r_i);$ 
5    $\text{depth}[r_i] \leftarrow 0; \text{depth}[v] \leftarrow \infty$  for all  $v \in V \setminus \{r_i\};$ 
6    $n = 0;$ 
7   while  $Q_L$  and  $Q_N$  are not empty do
8     for all  $u \in Q_L$  at depth  $n$  do
9       for all unvisited neighbors  $v$  of  $u$  do
10         $\text{depth}[v] \leftarrow n + 1;$ 
11        if  $v$  is a landmark then
12           $Q_N.\text{push}(v);$ 
13           $E_R \leftarrow E_R \cup \{(r_i, v)\};$ 
14           $\sigma(r_i, v) \leftarrow \text{depth}[v];$ 
15        else
16           $Q_L.\text{push}(v);$ 
17           $L(v) \leftarrow L(v) \cup \{(r_i, \text{depth}[v])\};$ 
18     for all  $u \in Q_N$  at depth  $n$  do
19       for all unvisited neighbors  $v$  of  $u$  do
20         $\text{depth}[v] \leftarrow n + 1;$ 
21         $Q_N.\text{push}(v);$ 
22      $n \leftarrow n + 1;$ 
```

Rysunek 1: Pseudokod procedury etykietowania

Algorithm 3: Computing a sketch S_{uv}

Input: $\mathcal{L} = (M, L)$, two vertices u and v .
Output: A sketch $S_{uv} = (V_S, E_S, \sigma_S)$

```
1  $V_S \leftarrow \emptyset, E_S \leftarrow \emptyset;$ 
2 for all  $\{r, r'\} \subseteq R$  do
3    $\pi_{rr'} \leftarrow +\infty;$ 
4   if  $(r, \delta_{ur}) \in L(u)$  and  $(r', \delta_{vr'}) \in L(v)$  then
5      $\pi_{rr'} \leftarrow \delta_{ur} + d_M(r, r') + \delta_{vr'};$ 
6  $d_{uv}^\top \leftarrow \min\{\pi_{rr'} \mid \{r, r'\} \subseteq R\};$ 
7 for all  $\{r, r'\} \subseteq R$  and  $\pi_{rr'} = d_{uv}^\top$  do
8    $E_S \leftarrow E_S \cup \{(u, r), (v, r')\};$ 
9    $\sigma_S(u, r) \leftarrow \delta_{ur}, \sigma_S(v, r') \leftarrow \delta_{vr'};$ 
10  for all  $(r_i, r_j)$  in the shortest path graph of  $(r, r')$  in  $M$  do
11     $E_S \leftarrow E_S \cup \{(r_i, r_j)\};$ 
12     $\sigma_S(r_i, r_j) \leftarrow \sigma(r_i, r_j);$ 
13  $V_S \leftarrow V(E_S);$ 
```

Rysunek 2: Pseudokod procedury szkicowania

2.2 Główne idee rozwiązań

3 Przegląd poprzednich rozwiązań

3.1 Klasyczne algorytmy

3.2 Algorytmy dokładne dla dużych grafów

3.3 Algorytmy aproksymacyjne

4 QbS

4.1 Idea

4.2 Przykład

4.3 Etykietowanie

Niech $G = (V, E)$ - graf, $R \subset V$ - zbiór punktów orientacyjnych spełniający $|R| \ll |V|$. Schemat QbS zaczyna się od przetwarzania wstępnego, którego celem jest znalezienie kompaktowej reprezentacji najkrótszych ścieżek pomiędzy punktami orientacyjnymi, zwanej meta-grafem grafu G . Następnie, bazując na tymże meta-grafie, tworzymy etykietowanie, każdemu wierzchołkowi przypisując każdemu wierzchołkowi taką etykietę, aby móc efektywnie policzyć szkic dla zapytania $SPG(u, v)$ dla dowolnych u i v .

Meta-graf

Meta grafem nazywamy trójkę $M = (R, E_R, \sigma)$, gdzie $E_R \subset R \times R$ jest zbiorem krawędzi postaci (r, r') spełniających warunek, że przynajmniej jedna najkrótsza ścieżka między r i r' w oryginalnym nie przechodzi przez żaden inny punkt orientacyjny. $\sigma : E_R \rightarrow \mathbb{N}$ przypisuje każdej krawędzi w E_R wagę równą długości najkrótszej ścieżki.

Algorithm 4: Searching on $G[V \setminus R]$

Input: $G^- = G[V \setminus R]$, S_{uv} , $\mathcal{L} = (M, L)$ **Output:** A shortest path graph G_{uv}

```
1  $d_{uv}^\top, d_u^*, d_v^* \leftarrow \text{get\_bound}(S_{uv});$ 
2  $P_u \leftarrow \emptyset, P_v \leftarrow \emptyset, d_u \leftarrow 0, d_v \leftarrow 0;$ 
3 Enqueue  $u$  to  $Q_u$  and  $v$  to  $Q_v$ ;
4  $\text{depth}_u[w] \leftarrow \infty, \text{depth}_v[w] \leftarrow \infty$  for all  $w \in V \setminus R$ ;
5  $\text{depth}_u[u] \leftarrow 0, \text{depth}_v[v] \leftarrow 0;$ 
6 while  $d_u + d_v < d_{uv}^\top$  do
7    $t \leftarrow \text{pick\_search}(P_u, P_v, d_u^*, d_v^*, d_u, d_v);$ 
8   if  $t = u$  then
9      $Q_u \leftarrow \text{forward\_search}(Q_u);$ 
10  if  $t = v$  then
11     $Q_v \leftarrow \text{backward\_search}(Q_v);$ 
12     $P_t \leftarrow P_t \cup Q_t; d_t \leftarrow d_t + 1;$ 
13     $\text{depth}_t[w] \leftarrow d_t$  for  $w \in Q_t$ ;
14    if  $P_u \cap P_v$  is not empty then
15      break;
16 if  $P_u \cap P_v \neq \emptyset$  then
17    $G_{uv}^- \leftarrow \text{reverse\_search}(P_u \cap P_v, G^-, \text{depth}_u, \text{depth}_v);$ 
18 if  $d_u + d_v = d_{uv}^\top$  then
19    $Z \leftarrow \emptyset;$ 
20   for all  $(r, t) \in E_S$  with  $t \in \{u, v\}$  do
21      $d_m \leftarrow \min\{\sigma_S(r, t) - 1, d_t\};$ 
22     for all  $w$  with  $\text{depth}_t[w] = d_m, (r, \delta_{wr}) \in L(w),$ 
23        $\delta_{wr} + d_m = \sigma_S(r, t)$  do
24          $Z \leftarrow Z \cup \{(w, r)\};$ 
25    $G_{uv}^\mathcal{L} \leftarrow \text{recover\_search}(S_{uv}, \mathcal{L}, Z, G^-, \text{depth}_u, \text{depth}_v);$ 
26  $G_{uv} \leftarrow G_{uv}^- \cup G_{uv}^\mathcal{L};$ 
```

Rysunek 3: Pseudokod przeszukiwania kierowanego szkicem

System etykietowania (ang. *Labelling scheme*)

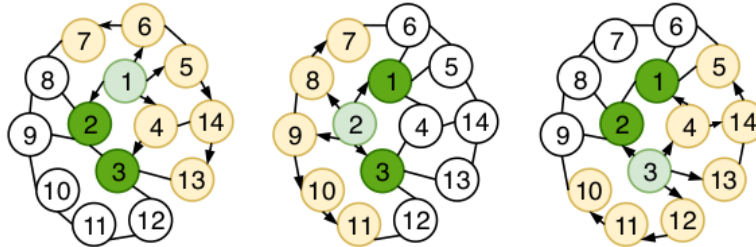
System etykietowania $\mathcal{L} = (M, L)$ składa się z meta-grafu M oraz etykietowania, które przypisuje każdemu wierzchołkowi $u \in V \setminus R$ etykietę $L(u)$ taką że:

$$L(u) = \{(r, \delta_{ur}) : r \in R, \delta_{ur} = d_G(u, r), (\exists p \in P_{ur})(V(p) \cap Rr)\}$$

Innymi słowy, para (r, δ_{ur}) jest częścią etykiety $L(u)$ tylko, jeśli istnieje co najmniej jedna najkrótsza ścieżka między u i r , która nie zawiera innych punktów orientacyjnych.

Algorytm etykietowania ukazano na obrazku 1. Wykonujemy BFS dla każdego punktu orientacyjnego r . Wykorzystujemy dwie kolejki do przechowywania odwiedzonych wierzchołków, zależnie od tego, czy mają one zostać zaetykietowane z wykorzystaniem r , czy też nie. Wszystkie wierzchołki oprócz r są na początku niezaetykietowane. Dla każdego wierzchołka w kolejce zaetykietowanych z r na n -tym poziomie BFS rozpatrujemy nieodwiedzonych

sąsiadów i oznaczamy ich jako odwiedzonych. Jeśli dany sąsiad jest punktem orientacyjnym to dodajemy krawędź do meta-grafu i ustawiamy głębokość jako dystans w funkcji σ . W przeciwnym przypadku dodajemy parę $(r, \text{depth}(v))$ do etykiety $L(V)$, a sam wierzchołek v do kolejki wierzchołków zaetykietowanych z r . Następnie rozpatrujemy kolejkę niezaetykietowanych na n -tym poziomie i wszystkich nieodwiedzonych sąsiadów dodajemy do kolejki niezaetykietowanych. Powtarzamy na kolejnych poziomach aż do przetworzenia wszystkich wierzchołków. Przykład ukazano na rysunku 4



Rysunek 4: Przykład etykietowania kolejno dla wierzchołków 1, 2 i 3

4.4 Szkicowanie grafu najkrótszych ścieżek

Aby móc efektywnie odpowiedzieć na zapytanie $SPG(u, v)$ niezbędne jest przygotowanie szkicu danych dla wierzchołków u i v . Będzie on oparty o schemat \mathcal{L} .

Szkic

Szkicem dla $SPG(u, v)$ na podstawie \mathcal{L} nazywamy trójkę $S_{uv} = (V_S, E_S, \sigma_S)$, gdzie $V_S = \{u, v\} \cup R$ jest zbiorem wierzchołków, E_S zbiorem krawędzi, a $\sigma_S \rightarrow \mathbb{N}$ zachowuje $\sigma_S(u', v') = d_G(u', v')$. Trójka ta zachowuje warunek, że E_S zawiera jedynie krawędzie na ścieżkach między u i v o minimalnej długości, a więc:

$$d_{uv}^T = \min_{(r, r')} \{ \delta_{ru} + d_M(r, r') + \delta_{r'v} : (r, \delta_{ru}) \in L(u), (r', \delta_{r'v}) \in L(v) \}$$

Oczywiście, zachodzi własność $d_{uv}^T \geq d_G(u, v)$. Algorytm ukazany na rysunku 3 tworzy omawiany szkic. Najpierw, zbiory wierzchołków i krawędzi w szkicu są puste. Zaczynamy od sprawdzenia wszystkich par punktów orientacyjnych, wyznaczając najkrótszą ścieżkę z u do v przechodzącą przez oba te punkty używając etykiet i meta-grafu. Najkrótsza otrzymana ścieżka to d_{uv}^T . Następnie, dla wszystkich wyznaczonych ścieżek o takiej długości, dodajemy ich krawędzie do zbioru krawędzi szkicu, łącznie z krawędziami łączącymi u i v z punktami obserwacyjnymi. Analogicznie, do zbioru dodajemy wszystkie wierzchołki na tych ścieżkach.

4.5 Przeszukiwanie kierowane

4.6 Poprawność

4.7 Złożoność pamięciowa

4.8 Złożoność czasowa (szkic)

4.9 Urównoleglenie

Łatwo pokazać, że dla danego zbioru punktów orientacyjnych R w G , istnieje dokładnie jeden system etykietowania \mathcal{L} spełniający definicję, jego wybór jest więc oczywiście deterministyczny. Pozwala to łatwe urównoleglenie algorytmu etykietowania.

4.10 Zależność od liczby punktów orientacyjnych

5 Zakończenie

Dziękuję słuchaczom za uwagę i życzę miłego dnia.