Politechnika Wrocławska

Algorytmy wyznaczania najkrótszych ścieżek w rzeczywistych sieciach drogowych

Autor: Tomasz Strzałka
Promotor: dr hab. Pawłeł Zieliński, prof. PWr
Wydział Podstawowych Problemów Techniki

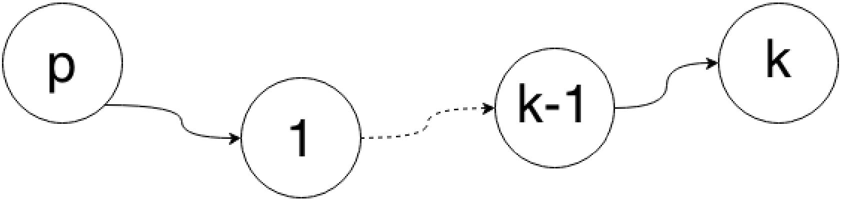
24 grudnia 2014

Problem najkrótszych ścieżek

Ścieżka

$$P = \langle v_p, v_1, \cdots, v_k \rangle \quad (1)$$

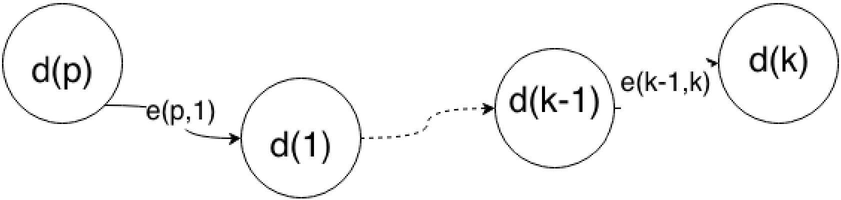
$$v_s \overset{k}{\rightsquigarrow} v_i \quad (2)$$



Problem najkrótszych ścieżek

Najkrótsza ścieżka

$$\sum_{e_{ij} \in P'} c_{ij} = minimum : e_{ij} \in P' \Leftrightarrow v_i, v_j \in P \wedge v_i \leadsto v_j = e_{ij} \ni E. \tag{3}$$



Problem najkrótszych ścieżek

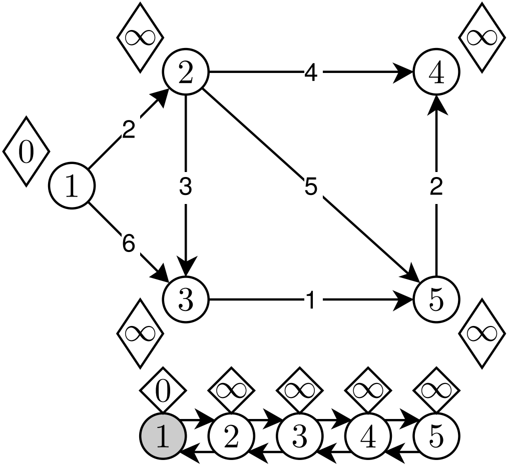
Najkrótsza ścieżka

$$d(i) = \begin{cases} \min\left\{c(s,i) : v_s \overset{*}{\rightsquigarrow} v_i\right\} & \text{jeśli } \exists v_s \overset{*}{\rightsquigarrow} v_i \\ \infty & \text{w przeciwnym przypadku} \end{cases} \tag{4}$$

gdzie:

$$c(p,k) = \sum_{e_{ij} \in P} c_{ij} \quad : \quad P = \langle v_p, v_1, \cdots, v_k \rangle \tag{5}$$

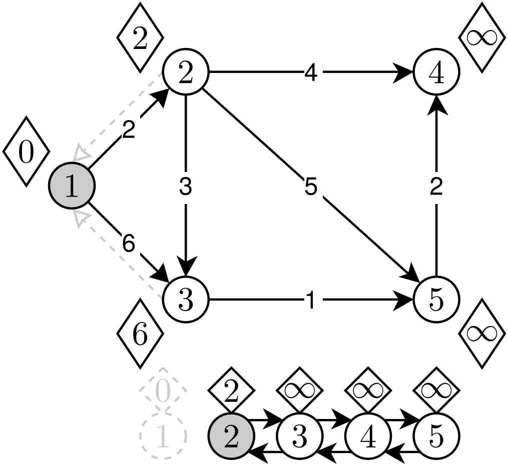
Algorytm Dijkstry



Algorytm Dijkstry

- 21:46:48 | INFO | [DKQD] PreExecution summary:
  Running algorithm : Dijkstra's Naive
  Implementation with double-linked lists
  Selected mode : Single Source
  Source node's ID : 1
- 21:46:48 | TRACE | [DKQD] Initialize double-linked list with source node with ID: 1 (distance: 0).
- 21:46:48 | TRACE | [DKQD] Query for next element from queue.

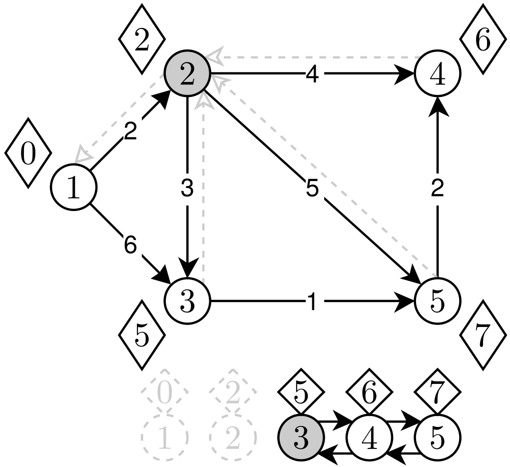
Algorytm Dijkstry



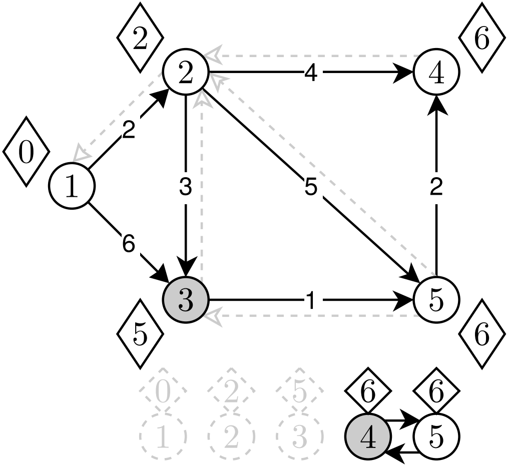
Algorytm Dijkstry

- 21:46:48 | TRACE | [DKQD] Queried node details:
  Node ID : 1
  Distance : 0
  Parent's ID: : no parent
- 21:46:48 | TRACE | [DKQD] Executing relaxation.
  Changing nodes' linkage from:
  No parents found for node with ID: 3 (4294967295) to:
  1 (0) —(6)—> 3 (6)
- 21:46:48 | TRACE | [DKQD] Executing relaxation.
  Changing nodes' linkage from:
  No parents found for node with ID: 2 (4294967295) to:
  1 (0) —(2)—> 2 (2)
  21:46:48 | TRACE | [DKQD] Query for next element

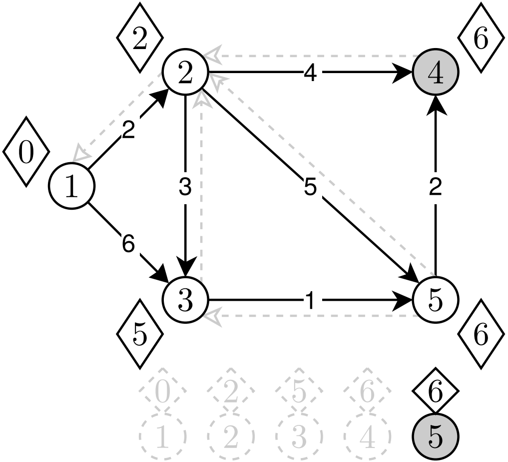
Algorytm Dijkstry



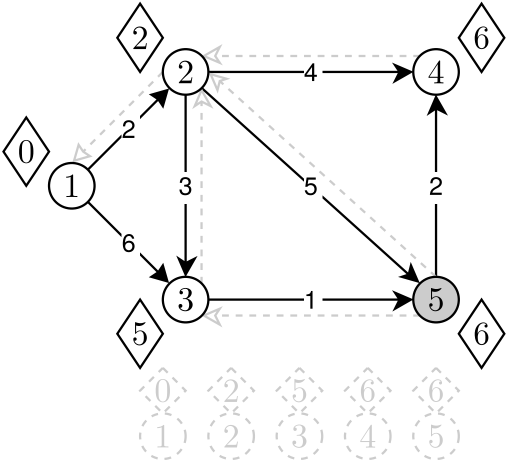
Algorytm Dijkstry



Algorytm Dijkstry



Algorytm Dijkstry



Algorytm Dijkstry

Algorithm 1: $DIJKSTRA(G, w, s)$

begin

G = ustaw $d(i) = \infty$ oraz $Pred(i) = NULL$ dla każdego węzła o $id = i$. Dodatkowo dla źródła $d(i) = 0$.

$S = \emptyset$

$Q = G.V$

while $Q \neq \emptyset$ **do**

  usuń z Q element o najniższym $d(i)$ i dodaj go do S

  for *każdy wierzchołek* $v \in A(i)$ **do**

    wykonaj relaksację

Wpływ struktury

na efektywność algorytmu

Złożoność (generyczny algorytm Dijkstry)

- inicjalizujemy $|V|$ wierzchołków,
- usuwamy łącznie $|V|$ wierzchołków,
- wykonujemy relaksację dla $|E|$ krawędzi (każdy wierzchołek odwiedzamy tylko raz),

Zatem:

- $|V| \cdot O(INIT(?))$,

$+\ |V| \cdot O(EXTRACT\_MIN(Q))$,

$+\ |E| \cdot O(UPDATE(?))$

Wpływ struktury

na efektywność algorytmu

Złożoność (generyczny algorytm Dijkstry)

- inicjalizujemy $|V|$ wierzchołków,
- usuwamy łącznie $|V|$ wierzchołków,
- wykonujemy relaksację dla $|E|$ krawędzi (każdy wierzchołek odwiedzamy tylko raz),

Zatem:

- $|V| \cdot O(INIT(?))$,

$+\ |V| \cdot O(EXTRACT\_MIN(Q))$,

$+\ |E| \cdot O(UPDATE(?))$

Wpływ struktury

na efektywność algorytmu

Złożoność (generyczny algorytm Dijkstry)

- inicjalizujemy $|V|$ wierzchołków,
- usuwamy łącznie $|V|$ wierzchołków,
- wykonujemy relaksację dla $|E|$ krawędzi (każdy wierzchołek odwiedzamy tylko raz),

Zatem:

- $|V| \cdot O(INIT(?))$,

$+ \; |V| \cdot O(EXTRACT\_MIN(Q))$,

$+ \; |E| \cdot O(UPDATE(?))$

Wpływ struktury

na efektywność algorytmu

Złożoność (generyczny algorytm Dijkstry)

- inicjalizujemy $|V|$ wierzchołków,
- usuwamy łącznie $|V|$ wierzchołków,
- wykonujemy relaksację dla $|E|$ krawędzi (każdy wierzchołek odwiedzamy tylko raz),

Zatem:

- $|V| \cdot O\left(INIT\left(?\right)\right)$,

$+$ $|V| \cdot O\left(EXTRACT\_MIN\left(Q\right)\right)$,

$+$ $|E| \cdot O\left(UPDATE\left(?\right)\right)$

INIT, UPDATE, EXTRACT

Implementacje Q

- Drzewa
- Kopce
- Listy
- Tablice
- Kubełki
- Stosy
- Łączenie struktur

INIT, UPDATE, EXTRACT

Implementacje Q

- Drzewa
- Kopce
- Listy
- Tablice
- Kubełki
- Stosy
- Łączenie struktur

INIT, UPDATE, EXTRACT

Implementacje Q

- Drzewa
- Kopce
- Listy
- Tablice
- Kubełki
- Stosy
- Łączenie struktur

INIT, UPDATE, EXTRACT

Implementacje Q

- Drzewa
- Kopce
- Listy
- Tablice
- Kubełki
- Stosy
- Łączenie struktur

INIT, UPDATE, EXTRACT

Implementacje Q

- Drzewa
- Kopce
- Listy
- Tablice
- Kubełki
- Stosy
- Łączenie struktur

INIT, UPDATE, EXTRACT

Implementacje Q

- Drzewa
- Kopce
- Listy
- Tablice
- Kubełki
- Stosy
- Łączenie struktur

INIT, UPDATE, EXTRACT

Implementacje Q

- Drzewa
- Kopce
- Listy
- Tablice
- Kubełki
- Stosy
- Łączenie struktur

Tablice, listy, stosy, kolejki, kubełki...

sposoby podejścia do problemu SSSP

Złożoność (generyczny algorytm Dijkstry)

$$|V| \cdot [O(INSERT(Struct, Node)) + O(EXTRACT\_MIN(Struct))] + |E| \cdot O(UPDATE\_STRUCT(Struct, Node))$$

	Tablica	Lista	Stos	k. R-arny	k. Fibonacciego
I	$O(n)$	$O(1)$	$O(1)$	$O(\log_R(n))$	$O(1)$
E	$O(n)$	$O(1)$	$O(1)$	$O(R \cdot \log_R(n))$	$O(\log(n))$
U	$O(n)$	$O(n)$	$O(n)$	$O(\log_R(n))$	$O(1)$

Iluzja matematyczna

Złożoność algorytmu Dijkstry - kopce R-arne

$|V| \cdot [\log_R(n) + R \cdot \log_R(n)] + |E| \cdot \log_R(n)$

	Tablica	Lista	Stos	k. R-arny	k. Fibonacciego
I	$O(n)$	$O(1)$	$O(1)$	$O(\log_R(n))$	$O(1)$
E	$O(n)$	$O(1)$	$O(1)$	$O(R \cdot \log_R(n))$	$O(\log(n))$
U	$O(n)$	$O(n)$	$O(n)$	$O(\log_R(n))$	$O(1)$

Iluzja matematyczna

Złożoność algorytmu Dijkstry - kopce R-arne

$|V| \cdot [\log_R(n) + R \cdot \log_R(n)] + |E| \cdot \log_R(n)$

Bardzo gęsty graf

$$|E| = \Omega\left(|V|^{1+\epsilon}\right) \tag{6}$$

$$R = |E| / |V| > 1 \tag{7}$$

$$|E| \cdot \log_R(n) = |E| \cdot \log(|V|) / \log(R) = |E| \cdot \log(|V|) / \log(|E| / |V|) \tag{8}$$

Iluzja matematyczna

Złożoność algorytmu Dijkstry - kopce R-arne

$|V| \cdot [\log_R(n) + R \cdot \log_R(n)] + |E| \cdot \log_R(n)$

Bardzo gęsty graf

$$|E| = \Omega\left(|V|^{1+\epsilon}\right) \tag{9}$$

$$|E| \cdot \log(|V|) / \log(|E| / |V|) = |E| \cdot \log(|V|) / \log\left(\left(|V|^{1+\epsilon}\right) / |V|\right) \tag{10}$$

$$= |E| \cdot \log(|V|) / \log(|V|^{\epsilon}|) = |E| \cdot \log(|V|) / (\epsilon \log(|V||)) \tag{11}$$

$$= |E| / (\epsilon) = |E| = m \tag{12}$$

Struktura sieci

Reprezentacje

- Macierz incydencji
  - węzeł-węzeł
  - węzeł-krawędź
- listy sąsiedztwa
  - pośrednie
  - bezpośrednie
- Star Representation
  - forward
  - reverse

Struktura sieci

Reprezentacje

- Macierz incydencji
  - węzeł-węzeł
  - węzeł-krawędź
- listy sąsiedztwa
  - pośrednie
  - bezpośrednie
- Star Representation
  - forward
  - reverse

Struktura sieci

Reprezentacje

- Macierz incydencji
  - węzeł-węzeł
  - węzeł-krawędź
- listy sąsiedztwa
  - pośrednie
  - bezpośrednie
- Star Representation
  - forward
  - reverse

Struktura sieci

Reprezentacje

- Macierz incydencji
  - węzeł-węzeł
  - węzeł-krawędź
- listy sąsiedztwa
  - pośrednie
  - bezpośrednie
- Star Representation
  - forward
  - reverse

Struktura sieci

Reprezentacje

- Macierz incydencji
  - węzeł-węzeł
  - węzeł-krawędź
- listy sąsiedztwa
  - pośrednie
  - bezpośrednie
- Star Representation
  - forward
  - reverse

Struktura sieci

Reprezentacje

- Macierz incydencji
  - węzeł-węzeł
  - węzeł-krawędź
- listy sąsiedztwa
  - pośrednie
  - bezpośrednie
- Star Representation
  - forward
  - reverse

Struktura sieci

Reprezentacje

- Macierz incydencji
  - węzeł-węzeł
  - węzeł-krawędź
- listy sąsiedztwa
  - pośrednie
  - bezpośrednie
- Star Representation
  - forward
  - reverse

Struktura sieci

Reprezentacje

- Macierz incydencji
  - węzeł-węzeł
  - węzeł-krawędź
- listy sąsiedztwa
  - pośrednie
  - bezpośrednie
- Star Representation
  - forward
  - reverse

Struktura sieci

Reprezentacje

- Macierz incydencji
  - węzeł-węzeł
  - węzeł-krawędź
- listy sąsiedztwa
  - pośrednie
  - bezpośrednie
- Star Representation
  - forward
  - reverse

Struktura sieci

Reprezentacje

- Macierz incydencji
  - węzeł-węzeł
  - węzeł-krawędź
- listy sąsiedztwa
  - pośrednie
  - bezpośrednie
- Star Representation
  - forward
  - reverse

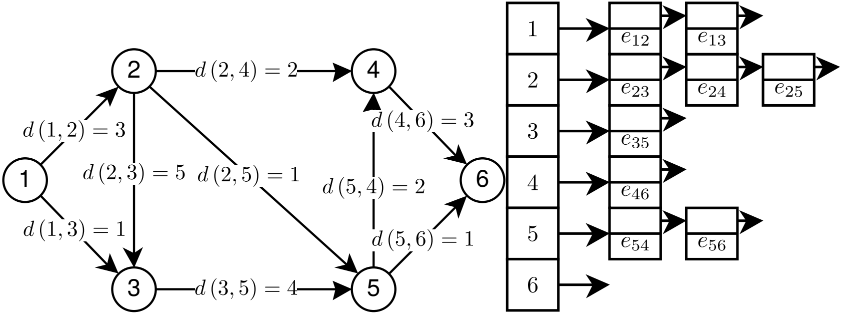
Reprezentacje grafu

	Macierze	
	Incydencji	Sąsiedztwa
Potrzebna pamięć	$O\left(\lvert V\rvert \cdot \lvert E\rvert\right)$	$O\left(\lvert V\rvert^2\right)$
Przegląd $v \in A\left(i\right)$	$O\left(\lvert E\rvert + \lvert A\left(i\right)\rvert \cdot \lvert V\rvert\right)$	$O\left(\lvert V\rvert\right)$
Dodawanie krawędzi	$O\left(1\right)$	$O\left(1\right)$
Dodawanie nowej krawędzi	$O\left(\lvert V\rvert \cdot \lvert E\rvert\right)$	$O\left(1\right)$
Usuwanie krawędzi	$O\left(\lvert V\rvert\right)$	$O\left(\lvert V\rvert\right)$
Trwałe usuwanie krawędzi	$O\left(\lvert V\rvert \cdot \lvert E\rvert\right)$	$O\left(\lvert V\rvert \cdot \lvert E\rvert\right)$
Stopień wierzchołka	$O\left(\lvert E\rvert\right)$	$O\left(\lvert V\rvert\right)$

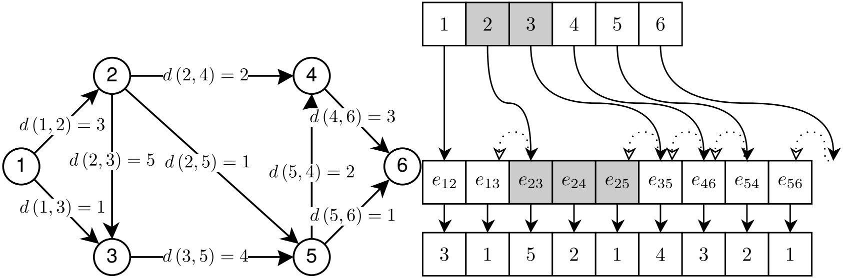
Reprezentacje grafu

	Listy Sąsiedztwa	Pęki Wejścia-wyjścia
Potrzebna pamięć	$O(\|V\|+\|E\|)$	$O(\|V\|+\|E\|)$
Przegląd $v \in A(i)$	$O(\|A(i)\|)$	$O(\|A(i)\|)$
Dodawanie krawędzi	$O(1)$	$O(\|V\|+\|E\|)$
Dodawanie nowej krawędzi	$O(1)$	$O(\|V\|+\|E\|)$
Usuwanie krawędzi	$O(\|E\|)$	$O(\|V\|+\|E\|)$
Trwałe usuwanie krawędzi	$O(\|E\|)$	$O(\|V\|+\|E\|)$
Stopień wierzchołka	$O(\|A(i)\|)$	$O(1)$

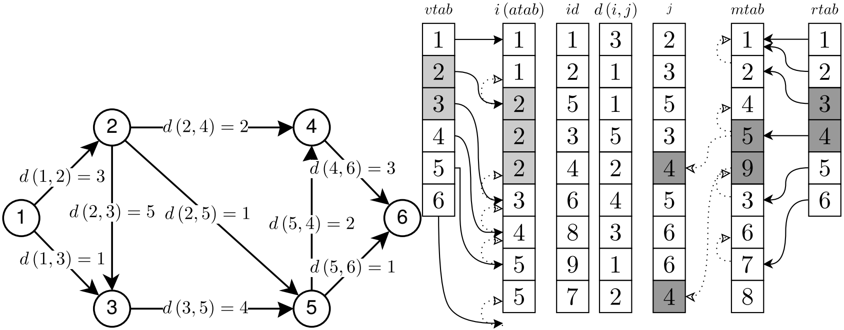
Listy sąsiedztwa



FSR


1
2
3
4
5
6
$d(1,2) = 3$
$d(1,3) = 1$
$d(2,3) = 5$
$d(2,4) = 2$
$d(2,5) = 1$
$d(3,5) = 4$
$d(4,6) = 3$
$d(5,4) = 2$
$d(5,6) = 1$
1 2 3 4 5 6
e_{12} e_{13} e_{23} e_{24} e_{25} e_{35} e_{46} e_{54} e_{56}
3 1 5 2 1 4 3 2 1

FRSR


d (1, 2) = 3
d (1, 3) = 1
d (2, 3) = 5
d (2, 4) = 2
d (2, 5) = 1
d (3, 5) = 4
d (4, 6) = 3
d (5, 4) = 2
d (5, 6) = 1
1
2
3
4
5
6
vtab
i (atab)
id
d (i, j)
j
mtab
rtab
vtab: 1 2 3 4 5 6
i (atab): 1 1 2 2 2 3 4 5 5
id: 1 2 5 3 4 6 8 9 7
d (i, j): 3 1 1 5 2 4 3 1 2
j: 2 3 5 3 4 5 6 6 4
mtab: 1 2 4 5 9 3 6 7 8
rtab: 1 2 3 4 5 6

Szybkość


Plot's title
Star Representation
Adjacency List
20000
15000
10000
5000
0
Y
0
500000
1e+06
1.5e+06
2e+06
2.5e+06
3e+06
X

Java

graphviz

algs4

perl

Biblioteka Take Me Home

API

Wymagania:

- pliki z danymi zgodne z formatem narzuconym podczas 9th DIMACS Implementation Challenge
- wierzchołki numerowane od 1

Podstawowe API

- logi oparte na idei biblioteki Log4j
- podstawowe struktury:
  - TMHConfig - konfiguracja algorytmów
  - TMH - konfiguracja biblioteki
- pseudo-obiektowość

Biblioteka Take Me Home

API

TMHConfig

- createTMHConfig(ścieżka do pliku z poleceniami)
  - createTMHConfig(USA-road-d.USA.ss);
- setAllowInterrupt(config,false);
- setCheckConfig(config,false);
- setGraphOrder(config,NONE);
  - NONE - z typu wyliczeniowego GraphOrder
- setGraphStruct(config,ADJACENCY_LIST);
  - ADJACENCY_LISTNE - z typu wyliczeniowego GraphStructAbbreviation

Biblioteka Take Me Home

API

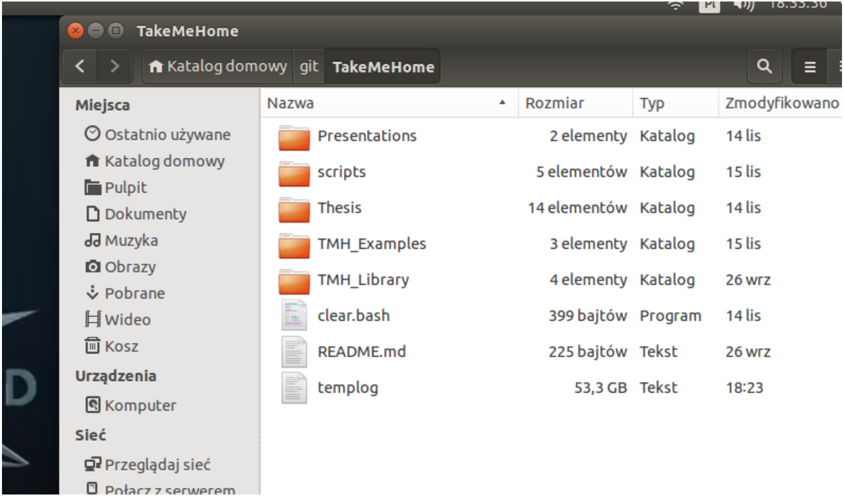
TMHConfig

- setAlgorithm(config,BFM);
  - BFM - z typu wyliczeniowego AlgorithmAbbreviation

TMH_API

- ins = createTMHAlgorithmInstance(config,"*.gr");
- runTMHAlgorithm(config->algorithm,ins);
- destroyTMHAlgorithmInstancje(alg,ins,false);
  - false - czy zresetować kofigurację

Ups


TakeMeHome
Katalog domowy
git
TakeMeHome
Miejsca
Ostatnio używane
Katalog domowy
Pulpit
Dokumenty
Muzyka
Obrazy
Pobrane
Wideo
Kosz
Urządzenia
Komputer
Sieć
Przeglądaj sieć
Nazwa
Rozmiar
Typ
Zmodyfikowano
Presentations 2 elementy Katalog 14 lis
scripts 5 elementów Katalog 15 lis
Thesis 14 elementów Katalog 14 lis
TMH_Examples 3 elementy Katalog 15 lis
TMH_Library 4 elementy Katalog 26 wrz
clear.bash 399 bajtów Program 14 lis
README.md 225 bajtów Tekst 26 wrz
templog 53,3 GB Tekst 18:23

Bibliografia

James B. Orlin

Network Flows: Theory, Algorithms, and Applications.

4. SHORTEST PATHS: LABEL-SETTING ALG.

Thomas H. Cormen

Wprowadzenie do algorytmów.

24. Najkrótsze ścieżki z jednym źródłem

F. Benjamin Zhan

Three Fastest Shortest Path Algorithms on Real Road Networks: Data Structures and Procedures.

Journal of Geographic Information and Decision Analysis, vol.1, no.1, pp. 70-82, 1997

Bibliografia

Warren B. Powell

A Generalized Threshold Algorithm for the SPP.

Department of Civil Engineering and Operations Research

Nishtha Kesswani

Design and Impl. of Multi-Parameter Dijkstra's Algorithm: A Shortest Path Alg. for Real-Road Networks.

International Journal of Advances in Engineering Research