Politechnika Wrocławska

Badanie efektywności algorytmu Bellmana-Forda w zależności od rozmiaru i gęstości grafu

Struktury Danych

Projekt 2

Hanna Grzebieluch, 264209

Termin zajęć: Wtorek, $17^{05} - 18^{35}$

Kod grupy: K00-35f

Wydział Informatyki i Telekomunikacji

Informatyczne Systemy Automatyki

13 czerwca 2023

Spis treści

1 Wstęp

Algorytm Bellmana-Forda służy do wyszukiwania najkrótszych ścieżek w grafie ważonym z wierzchołka źródłowego do wszystkich pozostałych wierzchołków.[1] W odróżnieniu od algorytmu Djikstry, dopuszcza ujemne wagi krawędzi. Algorytm może być także efektywnie używany do sprawdzenia występowania cykli ujemnych (ponieważ on sam ich nie dopuszcza).

2 Zasada działania algorytmu, lista sąsiedztwa

Algorytm dla każdego z wierzchołków przechowuje dwie wartości: koszt dotarcia do bieżącego wierzchołka **d** oraz wierzchołek poprzedni **p**. Dla wierzchołka startowego kosztem dotarcia jest 0, a poprzednim wierzchołkiem jest -1.
Algorytm analizowany jest na przykładzie implementacji listy sąsiedztwa. Początkowo inicjujemy wszystkiego komórki tablicy dotarcia na nieskończoność lub największą możliwą wartość - wyjątkiem jest komórka odwzorowująca wierzchołek startowy, gdyż w niej umieszczamy wartość 0. W komórkach tablicy dotyczących poprzedników umieszczamy natomiast wartość -1.
Całość powtarzamy **n-1** razy (gdzie przez n rozumiemy liczbę krawędzi), a następnie dokonujemy tzw. relaksacji krawędzi. Metoda relaksacji, która pomaga Nam wykryć ewentualne cykle ujemne, polega na sprawdzeniu, czy przy przejściu daną krawędzią grafu (u, v) o wadze k koszt dojścia jest większy od kosztu dojścia + k (k to waga), tj. dojście do wierzchołka v od wierzchołka u tą krawędzią jest tańsze od poprzednio znalezionych dojść. Jeśli natrafimy na taką krawędź, to ustawiamy koszt dojścia **d**[**v**] na koszt dojścia **d** [**u**] + **k** i w tablicy poprzedników dla **p**[**v**] umieszczamy numer wierzchołka u. Po wykonaniu n-1 obiegów, tablica d będzie zawierać koszty dojść do każdego z poszczególnych wierzchołków. Tablica p natomiast będzie dla każdego wierzchołka przechowywać jego poprzednika na wyszukanej ścieżce.
Aby poprawnie sprawdzić występowanie cykli ujemnych, należy ponownie przejrzeć zbiór krawędzi. W przypadku natrafienia na krawędź u–v o wadze k, dla której dalej koszt dojścia d[v] jest większy od d [u] = w, to mamy do czynienia z cyklem ujemnym. [2]

3 Badanie efektywności na różnych rodzajach grafów

3.1 Różne wielkości grafów

W celu zbadania efektywności algorytmu, dokonano pomiarów wyszukiwania najkrótszej ścieżki na grafach o odpowiednio 5, 10, 15 i 20 wierzchołkach.

```
Wierzcho|ek startowy: 0
Wierzcholek | Waga | Sciezka
0:        | 0 |   0
1:        | 3 |   0 1
2:        | -5 |   0 1 2
3:        | -7 |   0 1 3
4:        | 5 |   0 1 4


czas dla 5 wierzcholkow grafu: 0.002
```

Czas wykonania algorytmu dla grafu o 5 wierzchołkach

```
Wierzcho|ek startowy: 0
Wierzcholek | Waga | Sciezka
0:        | 0 |   0
1:        | 4 |   0 1
2:        | 10 |   0 1 3 2
3:        | 11 |   0 1 3
4:        | 9 |   0 1 4
5:        | 19 |   0 1 3 5
6:        | 14 |   0 1 3 6
7:        | 13 |   0 1 3 6 7
8:        | 23 |   0 1 3 5 8
9:        | 16 |   0 1 3 6 7 9


 czas dla 10 wierzcholkow grafu: 0.005
```

Czas wykonania algorytmu dla grafu o 10 wierzchołkach

```
Wierzcho|ek startowy: 0
Wierzcholek | Waga | Sciezka
0:        | 0 |   0
1:        | 3 |   0 1
2:        | 9 |   0 2
3:        | 4 |   0 1 4 6 3
4:        | 0 |   0 1 4
5:        | 11 |   0 1 4 6 3 5
6:        | 7 |   0 1 4 6
7:        | 14 |   0 1 4 6 11 9 8 7
8:        | 14 |   0 1 4 6 11 9 8
9:        | 10 |   0 1 4 6 11 9
10:        | 18 |   0 1 4 6 13 10
11:        | 10 |   0 1 4 6 11
12:        | 21 |   0 1 4 6 13 10 12
13:        | 13 |   0 1 4 6 13
14:        | 15 |   0 1 4 6 11 14


czas dla 15 wierzcholkow grafu: 0.005
```

Czas wykonania algorytmu dla grafu o 15 wierzchołkach

```
Wierzcholek | Waga | Sciezka
0:        | 0 |   0
1:        | 4 |   0 1
2:        | 0 |   0 3 5 7 4 2
3:        | 5 |   0 3
4:        | 2 |   0 3 5 7 4
5:        | 7 |   0 3 5
6:        | 8 |   0 3 5 7 6
7:        | 5 |   0 3 5 7
8:        | 15 |   0 3 5 7 6 8
9:        | 13 |   0 3 5 7 9
10:        | 11 |   0 3 5 7 6 8 10
11:        | 13 |   0 3 5 7 6 8 10 11
12:        | 10 |   0 3 5 7 6 8 10 11 12
13:        | 7 |   0 3 5 7 6 8 10 11 13
14:        | 13 |   0 3 5 7 6 8 10 11 12 14
15:        | 11 |   0 3 5 7 6 8 10 11 12 14 16 15
16:        | 9 |   0 3 5 7 6 8 10 11 12 14 16
17:        | 3 |   0 3 5 7 6 8 10 11 12 14 16 18 17
18:        | 8 |   0 3 5 7 6 8 10 11 12 14 16 18
19:        | 10 |   0 3 5 7 6 8 10 11 12 14 16 18 19


czas dla 20 wierzcholkow grafu: 0.006
```

Czas wykonania algorytmu dla grafu o 20 wierzchołkach

3.2 Zmiana wywołana poprzez zwiększenie liczby krawędzi

```
Wierzcho|ek startowy: 0
Wierzcholek | Waga | Sciezka
0:       | 0 |   0
1:       | 4 |   0 1
2:       | 10 |   0 1 3 2
3:       | 11 |   0 1 3
4:       | 9 |   0 1 4
5:       | 19 |   0 1 3 5
6:       | 14 |   0 1 3 6
7:       | 13 |   0 1 3 6 7
8:       | 23 |   0 1 3 5 8
9:       | 16 |   0 1 3 6 7 9

 czas dla 10 wierzcholkow grafu: 0.005
```

Czas wykonania algorytmu dla grafu o 10 wierzchołkach

```
Wierzcho|ek startowy: 0
Wierzcholek | Waga | Sciezka
0:       | 0 |   0
1:       | 4 |   0 1
2:       | 10 |   0 1 3 2
3:       | 11 |   0 1 3
4:       | 9 |   0 1 4
5:       | 18 |   0 1 5
6:       | 14 |   0 1 3 6
7:       | 13 |   0 1 3 6 7
8:       | 22 |   0 1 5 8
9:       | 16 |   0 1 9

czas dla 10 wierzcholkow grafu: 0.003
```

Czas wykonania algorytmu dla grafu o 10 wierzchołkach, ale większej liczbie krawędzi

Rysunek 3: Porównanie zmian w czasie dzialania po dodaniu większej liczby krawędzi do grafu o 10 wierzchołkach

```
Wierzcho|ek startowy: 0
Wierzcholek | Waga | Sciezka
0:       | 0 |   0
1:       | 3 |   0 1
2:       | 9 |   0 2
3:       | 4 |   0 1 4 6 3
4:       | 0 |   0 1 4
5:       | 11 |   0 1 4 6 3 5
6:       | 7 |   0 1 4 6
7:       | 14 |   0 1 4 6 11 9 8 7
8:       | 14 |   0 1 4 6 11 9 8
9:       | 10 |   0 1 4 6 11 9
10:       | 18 |   0 1 4 6 13 10
11:       | 10 |   0 1 4 6 11
12:       | 21 |   0 1 4 6 13 10 12
13:       | 13 |   0 1 4 6 13
14:       | 15 |   0 1 4 6 11 14

czas dla 15 wierzcholkow grafu: 0.005
```

Czas wykonania algorytmu dla grafu o 15 wierzchołkach

```
Wierzcho|ek startowy: 0
Wierzcholek | Waga | Sciezka
0:       | 0 |   0
1:       | 3 |   0 1
2:       | -9 |   0 1 13 10 8 7 2
3:       | -26 |   0 1 13 14 9 3
4:       | -31 |   0 1 13 14 9 3 4
5:       | -40 |   0 1 13 14 9 3 4 5
6:       | -41 |   0 1 13 14 9 3 4 5 6
7:       | -13 |   0 1 13 10 8 7
8:       | -13 |   0 1 13 10 8
9:       | -18 |   0 1 13 14 9
10:       | -15 |   0 1 13 10
11:       | -38 |   0 1 13 14 9 3 4 5 6 11
12:       | -12 |   0 1 13 10 12
13:       | -20 |   0 1 13
14:       | -12 |   0 1 13 14

czas dla 15 wierzcholkow grafu: 0.008
```

Czas wykonania algorytmu dla grafu o 15 wierzchołkach, ale większej liczbie krawędzi

Rysunek 4: Porównanie zmian w czasie dzialania po dodaniu większej liczby krawędzi do grafu o 10 wierzchołkach

3.3 Zestawienie pomiarów

l. wierzchołków	czas [ms]
5	0.002
10	0.005
15	0.005
20	0.006

Tabela 1: Czasy działania algorytmów dla wybranych ilości wierzchołków

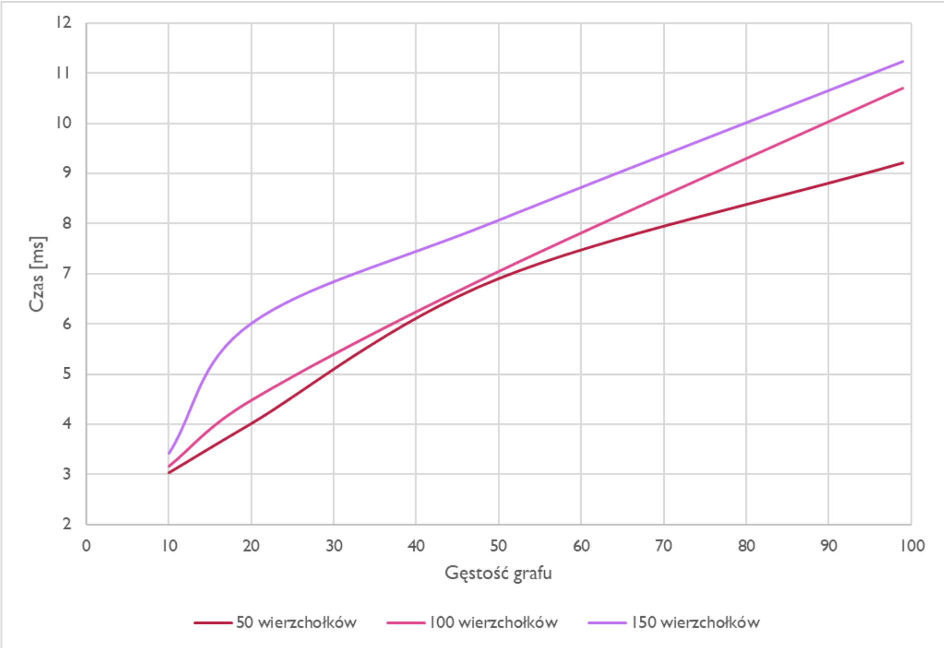
l. wierzchołków	l. krawędzi	czas [ms]
10	19	0.005
	35	0.003
15	30	0.005
	52	0.008

Tabela 2: Porównanie czasów działania algorytmów dla wybranych ilości wierzchołków i krawędzi

3.3.1 Generator grafów

Dla ulatwienia przeprowadzenia analizy wydajności algorytmu, w kodzie zawarty został generator losowych grafów, który pozwala na wygenerowanie grafu o porządanej liczbie wierzchołków i krawędzi (z pewnymi ograniczeniami - takimi jak minimalna i maksymalna liczba krawędzi).

Porównanie czasów działania algorytmu Bellmana-Forda dla różnych wielkości i gęstości grafu:



Rysunek 5: Czas działania algorytmu Bellmana-Forda dla różnych parametrów grafu

4 Wnioski

Jak widać na przedstawionych danych, czas działania algorytmu jest związany bezpośrednio z ilością wierzchołków i krawędzi w badanym grafie. Im graf ma więcej wierzchołków lub krawędzi, tym algorytm poświęca więcej czasu na jego przeszukanie. Algorytm Bellmana-Forda znajdowania najkrótszej ścieżki możnaby jednak usprawnić stosując m.in. kolejkę priorytetową.

Literatura

[1] "Algorytm bellmana-forda." https://pl.wikipedia.org/wiki/Algorytm_Bellmana-Forda. data dostępu: 17.05.2023.

[2] "Najkrótsza ścieżka w grafie ważonym, algorytm bellmana-forda." https://eduinf.waw.pl/inf/alg/001_search/0138a.php. data dostępu: 19.05.2023.