Iga Słotwińska

Zawory

Opis problemu
Aby znaleźć najmniejszy podzbiór zaworów, które odetną grupę wybranych wierzchołków od dopływu do wody, sprowadzam problem do postaci sieci przepływowej i szukania minimalnego przecięcia w grafie.

1. Sieć przepływowa - pomocniczy graf w postaci sieci przepływowej, na którym wykonywane są oba algorytmy
    1. Źródła tworzą 1 wierzchołek S, wielokrotne krawędzie są usuwane
    2. Wierzchołki do usunięcia tworzą 1 wierzchołek T, wielokrotne krawędzie są usuwane
    3. Wierzchołki i krawędzie (o nieskończenie dużej przepustowości) są dodawane do sieci, idąc rekurencyjnie przez krawędzie wychodzące ze źródła
    4. Wierzchołki bez krawędzi wychodzących są usuwane
2.
    1. Maksymalny przepływ -
        1. Wierzchołki będące zaworami dzielone są na 2 wierzchołki połączone 1 krawędzią o przepustowości 1
        2. Algorytm Dinica - maksymalny przepływ mówi o minimalnej liczbie zaworów potrzebnych do odcięcia spływu(o złożoności wierzchołki^2*krawędzie)
        3. Zawory, do których da się dotrzeć po ostatniej iteracji, po usunięciu zaworów nadmiarowych, są rozwiązaniem. Nadmiarowe zawory istnieją, jeśli liczność zbioru zaworów do usunięcia jest większa od obliczonego maksymalnego przepływu.
            1. Nadmiarowe zawory są najpierw usuwane, poprzez sprawdzenie przeszukiwaniem w głąb, czy wszystkie jego dzieci należą do podzbioru zaworów do usunięcia. Jeśli tak, jest on usuwany.
            2. Inną możliwością byłoby szukanie podzbiorów - na pewno prawidłową, o liczności równej maksymalnemu przepływowi, wśród wszystkich zaworów, jednak znacznie rośnie wtedy złożoność (symbol newtona: zawory nad maksymalny przepływ).
    2. Brute force - przeglądanie wszystkich możliwych podzbiorów zaworów o złożoności 2^n
        1. Binarny ciąg reprezentujący zawory (ograniczenie górne do 63 zaworów w sieci przepływowej)
        2. Sprawdzanie czy dany zbiór zaworów rozcina graf (zaczynając od podzbioru n-1 elementowego, gdzie n jest liczbą zaworów), jeśli tak, próba znalezienia mniejszego podzbioru o takiej własności
        3. Najmniejszy znaleziony zbiór jest rozwiązaniem

Należy zauważyć, że problem będzie skuteczny w przypadku dużej ilości źródeł i wierzchołków do odcięcia w stosunku do pozostałych wielkości. Przy zmianie grafu na sieć przepływową kilkukrotnie można zmniejszyć rząd wielkości problemu. Algorytm będzie nieefektywny w przypadku grafów zbliżonych do sieci przepływowych, wtedy wszelka modyfikacja wniesie niewiele poprawek. Najsłabszym punktem proponowanego rozwiązania jest ostatni etap szukania podzbiorów, w najgorszym przypadku będzie bliski rozwiązaniu brute force.

Tryby wykonania programu
- -m1 < filename.txt (czytanie grafu z pliku)
- -m2 n m (generowanie grafu: n - liczba zaworów w grafie, m - współczynnik gęstości grafu)
- -m3 m n o p (generowanie wielu przypadków grafów: n - liczba skoków, m - początkowa liczba zaworów w grafie, o - wielkość skoku liczby zaworów, p - liczba powtórzeń dla każdej wielkości n)

Wejście
Wejściem jest graf w postaci:
(Nr wierzchołka) (typ)
-
(wierzchołek początkowy krawędzi) (wierzchołek końcowy krawędzi)
-
(Nr wierzchołka do odcięcia od źródła) (kolejne wierzchołki)

Typ wierzchołka:

1. Źródło - s
2. Odpływ - l
3. Zawór - v

Wyjście
W zależności od trybu programu wyjściem jest zbiór zaworów, które trzeba zakręcić aby odciąć od źródła wybrany podzbiór wierzchołków i czas wykonania algorytmu lub tabela porównująca złożoność i czasy wykonania dla różnych wielkości problemu.

Program
1. Main - główna klasa, sterująca trybem wykonywania i pomiarami czasów
2. Parser - klasa czytająca graf z pliku
3. graph - pakiet, w którym znajdują się klasy odpowiedzialne za graf przed modyfikacją go w sieć przepływową
4. flowNetwork - pakiet odpowiadający pakietowi graph, dotyczy sieci przepływowej
    1. FlowNode - wierzchołek mający zbiory krawędzi wejściowych i wyjściowych, swoje id, typ oraz przypisany poziom przydatny podczas sprawdzania maksymalnego przepływu oraz sprawdzaniu, czy graf jest spójny.
    2. FlowEdge - krawędź, zawiera wierzchołek wejściowy i wyjściowy
    3. FlowNetwork - zawiera krawędzie grafu, wierzchołki, spływ oraz źródło
5. Generator - generator grafu. Funkcja tworząca graf przyjmuje liczbę źródeł, zaworów, odpływów, krawędzi i wierzchołków do usunięcia. Źródła mają losowo przydzielane 2.5% krawędzi, następnie pozostała połowa krawędzi jest losowana wśród połowy zaworów i odpływów. Następnie dodane zostają wierzchołki do odcięcia i także spływa do nich 2.5% krawędzi. Pozostałe krawędzie są losowo przydzielane. Ograniczenie na liczbę krawędzi spływającą i wypływającą ze źródeł/odcinanych wierzchołków wynika z faktu, że w przeciwnym wypadku generowany jest graf z podzbiorem wierzchołków, których nie da się odciąć od źródła.
6. Algorithms - zaimplementowane algorytmy
    1. BF - aby uniknąć powielania się danych, w całym projekcie kolekcje są zbiorami, jednak podczas przeszukiwania podzbiorów operuję na liście w celu prostszego przemieszczania się po jej elementach.

Testy
Testy zostały wykonane w środowisku MacOS, procesorze 2,6 GHz Intel Core i7 i pamięci RAM 16GB.

- Testy porównujące złożoność wykonałam jedynie dla heurystycznego rozwiązania, ponieważ algorytm bruteforce wykonuje się zbyt długo i nie zadziała dla grafów o liczbie zaworów w sieci przepływowej większej od 63. Przy szukaniu podzbiorów (pomimo zmniejszenia rozmiaru) algorytm ma bardzo różną złożoność, zależną nie tylko od ilości zaworów, ale także od liczby wszystkich wierzchołków, liczby krawędzi oraz maksymalnego przepływu.

```
//dla złożoności edges*vertices^2
n     |vertices              |edges                 |t(n)                  |q(n)
20    |24,00                 |44,23                 |5,67                  |3,36
30    |35,71                 |95,07                 |2,29                  |0,54
40    |49,14                 |106,00                |3,43                  |0,40
50    |59,14                 |143,10                |1,57                  |0,12
60    |76,38                 |163,80                |21,25                 |0,91
70    |85,80                 |188,29                |32,00                 |1,06
80    |99,50                 |202,30                |11,50                 |0,27
90    |109,17                |241,45                |815,17                |15,82
100   |121,80                |290,20                |12,60                 |0,19
110   |137,80                |315,60                |98,80                 |1,15

n     |vertices              |edges                 |t(n)                  |q(n)
20    |24,00                 |63,00                 |6,00                  |1,54
30    |34,00                 |88,50                 |5,00                  |1,27
40    |51,67                 |113,67                |5,00                  |1,08
50    |61,00                 |124,67                |5,00                  |1,00
60    |74,50                 |150,50                |16,50                 |3,26
```

70	87,75	203,00	2,00	0,45
80	97,75	246,25	242,25	59,72

n	vertices	edges	t(n)	q(n)
20	55,00	85,00	4,00	4,26
30	83,67	94,33	1,33	0,68
40	115,50	154,50	2,50	1,10
50	111,00	162,50	2,00	1,00
60	182,00	156,00	1,00	0,18
70	206,00	233,00	6,00	1,25
80	214,00	295,00	3,33	0,81

n	vertices	edges	t(n)	q(n)
20	86,00	97,00	4,00	5,90
30	123,00	119,50	5,00	4,44
40	157,50	156,75	1,75	1,24
50	212,67	201,00	2,00	1,00
60	235,00	250,50	2,00	1,02
70	312,00	325,00	1,50	0,56
80	335,50	336,50	2,00	0,67

n	vertices	edges	t(n)	q(n)
20	120,00	149,50	7,75	7,38
30	163,00	165,00	3,00	1,71
40	217,00	233,00	2,00	0,91
50	327,00	388,50	3,00	1,00
60	358,00	384,67	2,67	0,73
70	404,00	354,00	2,00	0,40
80	484,00	452,00	1,50	0,27

- Porównanie jakości rozwiązań dla „szybszego” algorytmu heurystycznego. Otrzymane rozwiązania nie zawsze są optymalne, jednak odcinają wybrane wierzchołki od źródeł.

```
Valves = 30
Density = 6
Dinic: 58 ms
{ 32 35 42 }
Max flow: 3
Nodes to cut are reachable: false
BF:    368 ms
{ 32 35 42 }

Dinic: 89 ms
{ 48 33 38 24 26 27 28 44 46 31 }
Max flow: 4
Nodes to cut are reachable: false
BF: zbyt długi czas oczekiwania
```

- Dla niewielkich grafów rozwiązanie bruteforce jest zdecydowanie szybsze.

```
Valves = 4
Vertices(all) = 6
MAX FLOW 1
Dinic: 31 ms
{ 2 4 }
BF:    9 ms
{ 2 3 }
```