Zaawansowane Techniki Programowania
Praca domowa 1
Dominik Freicher
19.10.2016

OPIS ZADANIA

Zadanie polega na wyznaczeniu ścieżki o najmniejszym koszcie transportu łączącą wierzchołek o numerze 1 z wierzchołkiem k, oraz wyznaczyć wartość tego kosztu. Koszt transportu wzdłuż każdej krawędzi ścieżki równy jest odwrotności przepustowości tej krawędzi. Koszt transportu dla każdego z węzłów ścieżki (dla węzła początkowego oraz końcowego z definicji przyjmujemy 0) definiowany jest jako bezwzględna wartość różnicy przepustowości krawędzi ścieżki o końcach w tym wierzchołku. Łączny koszt transportu wzdłuż ścieżki definiowany jest jako suma kosztów transportu wszystkich krawędzi wchodzących w skład ścieżki oraz wierzchołków leżących na ścieżce.

ZASTOSOWANY ALGORYTM

Podczas implementacji rozwiązania problemu użyty został zmodyfikowany algorytm przeszukiwana w głąb. Główną modyfikacją zastosowaną podczas implementacji algorytmu jest parametryzacja wierzchołka startu i końca. W pierwszej fazie algorytm znajduje wszystkie możliwe ścieżki pomiędzy zadanymi wierzchołkami. Następnie dla każdej ścieżki określa koszt transportu i na tej podstawie wybierana jest ścieżka o najmniejszym koszcie transportu. Zgodnie z założeniem wierzchołek startu określony jest jako 1, natomiast wierzchołek końca ścieżki przekazywany jest jako parametr.

STRUKTURY DANYCH

Projekt składa się z dwóch klas `Client.java` oraz `Path.java`. Poniżej przedstawione zostały najważniejsze elementy wchodzące w skład wymienionych klas.

```
Client
   main(String[] args)
   initDatabaseConnection(final String url)
   fetchingData(final ResultSet result)
   searchBestPath(int startPoint, int endPoint)

Path
   process()
   searchPath(int endPoint, List<Integer> road)
   removeRoadData(List<Integer> road)
   addRoadData(List<Integer> road, int id, int y)
   getEdgesOfVertex(int id)
   calculateTotalCost(float cost)
   printResult()
   Edge
```

Definicja programu głównego zawarta została w klasie `Client`. W metodzie **`main`** zrealizowana została logika odpowiadająca za pobranie argumentów oraz wywołanie metody nawiązującej połączenie z bazą danych **`initDatabaseConnection`**. Metoda **`fetchingData`** realizuje operacje pobrania danych z repozytorium. Metoda **`searchBestPath`** inicjalizuje operację uruchomienia algorytmu głównego.

Implementacja głównego algorytmu zawarta została w klasie **`Path`**. W klasie tej zdefiniowana została struktura pomocnicza w postaci klasy Edge, która symbolizuje pojedynczą krawędź grafu. W konstruktorze klasy **`Path`** przekazane zostają punkt startowy, punkt końcowy oraz lista krawędzi grafu w raz z ich wagami. W metodzie **process** ustawione zostają wartości początkowe dla algorytmu oraz wywołana zostaje metoda **searchPath**, której zadaniem jest znalezienie ścieżki według zadanych parametrów punktu startowego oraz końcowego. Metoda ta działa rekurencyjnie. Zadaniem metody **getEdgesOfVertex** jest znalezienie krawędzi łączących się z danym wierzchołkiem. Całkowity koszt transportu ścieżki wyliczany jest w metodzie **calculateTotalCost**. Metoda **printResult** odpowiada za wyświetlenie wyniku końcowego.

URUCHOMIENIE

Uruchomienie programu możliwe jest za pośrednictwem linii komend:

```
java Client <connection_string> <indeks>
```

Jako argumenty należy przekazać adres połączenia do bazy danych oraz indeks wierzchołka celu.