Arkadiusz Michalak - 291092
Projekt indywidualny - Sprawozdanie Końcowe
29.11.2018

Spis treści

Arkadiusz Michalak - 291092
Projekt indywidualny - Sprawozdanie Końcowe
29.11.2018

1 Wstęp

Dokument został stworzony w celu podsumowanie efektów prac nad projektem. Celem projektu było napisanie programu który dla podanych walut i kursów obliczy i wskaże najbardziej opłacalną ścieżkę wymiany. Drugą funkcjonalnością programu jest znalezienie i wskazanie arbitrażu ekonomicznego dla dowolonej waluty w zbiorze.

2 Opis algorytmu

Kluczowym elementem programu jest sam algorytm sprawdzania utworzonego grafu, Graf tworzą wierzchołki odpowiadające walutom oraz krawędzie odpowiadające kursom. Sam sposób działania algorytmu opiera się w dużej mierze na przeszukiwania grafu w szerz BFS - **link**.

2.1 Wyszukiwanie arbitrażu

Algorytm zaczyna od pierwszego wierzchołka na liście wstawia wartość podaną przez użytkowanika i zaczyna przeglądać jego sąsiadów. Każdy wierzchołek który otrzymuje wyższą wartość trafia do kolejki i jeszcze raz sprawdzane jest jego sąsiedctwo. Sytuacja dla której potwierdzane jest znalezienie arbitrażu wygląda następująco: W wierzchołku od którego zaczynaliśmy wartość może zostać zmieniona na wyższą, jeśli zachodzi taka sytuacja to wychodzimy z metody i funkcja zapisu zaczynając od tego wierzchołka wypisuje jego przodków. Jeśli taka sytuacja nie zaszła powtarzamy algorytm aż do przejścia całej listy wierzchołków.

2.2 Wyszukiwanie najlepszej wymiany

Algorytm zaczyna działanie od wyszukania wierzchołka waluty wejściowej, wstawia wartość podaną przez użytkownika. Wierzchołek jest dodawany do kolejki i rozpoczyna się działanie. Sprawdzani są wszyscy sąsiedzi wierzchołka zdejmowane z kolejki. Każdy wierzchołek którego wartość wzrosła jest dodawany do kolejki. Algorytm nie pozwala na zwiększanie wartości przez powtórzenie cyklu. Wyklucza to zapętlanie się do nieskończoności. Działanie algorytmu kończy się po opróżnieniu kolejki. Następnie odczytywana jest ścieżka przodków dla wierzchołka waluty wyjściowej.

3 Krótka instrukcja obsługi

Program należy wywołać przy użyciu narzędzia Power Shell znajdując się w folderze z plikiem waluty.jar *java -jar waluty.jar*. W tym samym folderze co plik jar musi znajdować się folder data a w nim plik danych wejściowych w formacie txt, jako argumenty wywołania należy podać:

1. arbitraz lub wymiana
2. nazwe pliku
3. ilość waluty
4. jeśli wymiana to walute wejściowa
5. jeśli wymiana to walute wyjściową

4 Efekty działania programu

W tym rozdziale zostaną zaprezentowane odpowiedzi programu na wywołanie zarówno prawidłowe jak i błedne. Prezentowane będzie polecenie jakie wywołano oraz wyświetlone przez programu wiadomości. Pliki które były używane do testowania znaleźć można w folderze data znajdującym się w głównym katalogu projektu.

4.1 Działanie prawidłowe

- java -jar waluty.jar arbitaz Test.txt 10000
  Rozpoczynam liczenie arbirazu
  Wynik: 10001,9614 EUR
  Ścieżka: EUR GBP USD EUR

- java -jar waluty.jar wymiana TestWydajnosci.txt 1000 USD RUB
  Rozpoczynam liczenie wymiany
  Wynik: 66241,7073 RUB
  Ścieżka: USD EUR GBP SOS XBT PLN INR RUB

- java -jar waluty.jar wymiana TestWieluCykli.txt 1000 USD JPY
  Rozpoczynam liczenie wymiany
  Wynik: 1331,0000 JPY
  Ścieżka: USD EUR GBP JPY

4.2 Działanie przy błednych danych

- java -jar waluty.jar arbitaz TestBrakuPolaczen.txt 10000
  Rozpoczynam liczenie arbirazu
  Ścieżka: Nie istnieje arbitraz

- java -jar waluty.jar wymiana TestBrakuPolaczen.txt 1000 EUR USD
  Rozpoczynam liczenie wymiany
  Ścieżka: Podana waluty nie sa polaczone

- java -jar waluty.jar wymiana TestBrakuPolaczen.txt 1000 EUR No
  Rozpoczynam liczenie wymiany
  Ścieżka: Podana waluta wyjściowa nie istnieje

5 Wprowadzone zmiany

1. Metode sprawdzającą sąsiadów w grafie rozbito na dwie różne od siebie metody, pierwsza sprawdza sąsiadów szukając arbitrażu, druga szukając najlepszej wymiany, różnią się warunkami wyjścia z metody oraz zwracanymi wartościami. Odpowiadają im dwie metody klasy **Graph** które dalej przetwarzają zwrócone informacje. Zmiana ta była motywowana znaczną różnicą pomiedzy tym co musi zostać sprawdzone dla znalezienia arbitrażu. Analogicznie do tego jak działa algorytm, opisany w rozdziale 2.

2. Znacznie ograniczono planowaną liczbę wyjątków jakie zwracać będzie program. Powodem tego było fakt, że większość błedów w danych czy parametrach wywołania uniemożliwia wykonanie programu. To też planowane wyjątki zastąpiono wypisem komunikatu i zamknięciem programu.

6 Podsumowanie i wnioski końcowe

Tworzenie programu oraz samo rozwiązanie problemu teoretycznie było zadaniem adekwatnie zajmującym i rozwiającym. Zakończyło się sukcesem i powstaniem sprawnie działającego programu. Wprowadzono znaczną ilość ważnych zmian względem założeń zawartych w specyfikacjach. Wnioski jakie

zostały wyciągnięte w czasie pracy nad projektem przedstawiają się następująco:

1. Dobrą decyzja był wybór BFS jako bazowego algorytmu i modyfikowanie go tak by spełnić założenia.

2. Zrezygnowano z tworzenie parsera plików tekstowych uniwersalnego zastosowania ze względu na to, że stworzenie go z funkcji dostępnych w standardowej biblotece Java przebiegło bardzo sprawnie.

3. Przyjętą dobrą strukture podziału programu na pakiety i klasy, pomogło to usprawnić prace nad projektem w etapie implementacji.