Praca domowa 1 – levenshtein

Przemysław Kleszcz

Zadanie

Cel pracy domowej: Wykorzystując odległość Levenshteina, odnaleźć w pliku danych wejściowych, łańcuch tekstowy najbardziej zbliżony do podanego wzorca.

Organizacja strukturalna

Projekt zawiera dwa pliki: **AppClient.java** oraz **Levenshtein.java**

AppClient.java:
public class AppClient
public static void main(String[] args)
private static BufferedReader getFileBufferedReader(String path)
private static int getLowestDistance(BufferedReader reader, String testString)

Levenshtein.java:
public class Levenshtein
public int getDistance(CharSequence testString, CharSequence line)
private int[] compare(CharSequence testString, CharSequence line, int lowerLength, int higherLength)

main(String[] args) - Przyjmuje jako parametr tablicę argumentów linii komend. Punkt wejścia programu.

getFileBufferedReader(String path) - Przyjmuje jako parametr ścieżkę do pliku danych testowych. Zwraca obiekt *BufferedReader* pozwalający na odczyt linii tekstu w pliku w sposób sekwencyjny.

getLowestDistance(BufferedReader reader, String testString) – Przyjmuje jako parametry obiekt *BufferedReader* oraz ciąg znaków, będący wzorcem dla pliku danych testowych. Zwraca wartość typu integer, oznaczającą numer linii dla której otrzymano najmniejszą wartość edycyjną.

getDistance(CharSequence testString, CharSequence line) – Przyjmuje jako parametry ciąg znaków wzorca oraz ciąg znaków pojedynczej linii pliku danych testowych. Zwraca wartość typu integer, oznaczającą wartość edycyjną dla wzorca względem pojedynczej linii.

compare(CharSequence testString, CharSequence line, int lowerLength, int higherLength) - Przyjmuje jako parametry ciąg znaków wzorca, ciąg znaków pojedynczej linii pliku danych testowych, długość mniejszego ciągu znaków, długość większego ciągu. Zwraca tablicę, będącą wynikiem przeliczeń zmodyfikowanej wersji algorytmu Levenshteina.

Opis mechanizmu

Program pobiera dwa parametry z linii komend. Pierwszym parametrem jest ścieżka do pliku danych testowych. Drugi parametr to ciąg znaków wzorca. Dane pliku są czytane sekwencyjnie w pętli, linia po linii. W każdym kroku pętli obliczana jest wartość edycyjna dla wzorca oraz pojedynczej linii tekstu oraz zapamiętywany jest indeks linii o najmniejszej wartości edycyjnej. Program kończy się wraz z wykonaniem wszystkich kroków pętli lub w przypadku odnalezienia wartości edycyjnej równej 0.

Algorytm obliczania wartości edycyjnej jest zmodyfikowaną wersją algorytmu Levenshteina. W pierwszej kolejności pobieramy długość ciągu wzorca i pojedynczej linii pliku. Sprawdzamy przypadek szczególny – jeśli długość wzorca wynosi 0 to wartością edycyjną jest długość pojedynczej linii. Jeśli długość pojedynczej linii wynosi 0 to wartością edycyjną jest długość wzorca. Jeśli długość wzorca jest większa od długości linii – wzorzec oraz linię zamieniamy referencjami. Tworzymy dwie jednowymiarowe tablice o długości mniejszego z dwóch ciągów znaków + 1. Jedna tablica zawiera bieżące wartości przeliczeń algorytmu, druga służy jako uchwyt do poprzedniego wiersza dla macierzy algorytmu Levenshteina. Na początku tablica poprzedniego wiersza wypełniana jest indeksami poszczególnych elementów. Następnie przebiega iteracja po wszystkich znakach jednego z ciągów. W każdym kroku, pierwszy element tablicy bieżących przeliczeń algorytmu jest ustawiany na wartość indeksu pętli. Bieżący znak jest zapamiętywany w zmiennej. Wewnątrz pętli przebiega następna iteracja po wszystkich znakach drugiego z ciągów. Obliczany jest koszt – jeśli bieżący znak z pętli wewnętrznej jest równy bieżącemu znakowi pętli zewnętrznej, koszt przyjmuje wartość 0, w przeciwnym wypadku 1. Element bieżącej tablicy przeliczeń o indeksie wewnętrznej iteracji jest ustawiany wartością będącą minimum z (gdzie *i* to indeks wewnętrznej iteracji):

- Wartości elementu bieżącej tablicy przeliczeń [*i* – 1] + 1

- Wartości elementu tablicy poprzedniego wiersza [*i*] + 1

- Wartości elementu tablicy poprzedniego wiersza [*i* – 1] + *koszt*

Po skończeniu iteracji wewnętrznej tablica bieżących wartości oraz tablica poprzednich wartości są zamieniane referencjami. Wstawianie wartości jest kontynuowane aż do zakończenia pętli zewnętrznej. Wartością edycyjną jest wartość elementu ostatniej uzupełnionej tablicy jednowymiarowej o najwyższym indeksie.