Системи, основани на знания

Документация

/Домашна работа (1)/

Изготвил: *Катерина Симеонова, ИС, група 1*

1.Описание на решението.

За решение на задачата е използван алгоритъмът на Дейкстра за намиране на най-оптимален път в дадения граф, понеже в контекста на конкретната задача той е най-подходящ в сравнение с други алгоритми за обхождане на граф.

В зависимост от имплементацията сложността на алгоритъма варира между O(E + V log(V)) и O(V^2), където:

E - броят на върховете
V - броят на ребрата

2.Структура на задачата.

Входните данни се четат от CSV файл, който съдържа представянето на графа като матрица на съседство. След като бъдат прочетени от файла, от данните се конструира матрица,която представлява графа и от там се пуска алгоритъмът на Дийкстра с предварително подадени параметри за граф, начална и крайна точка.

Класът Graph има следните полета:

```
class Graph{
        List<List<Integer>> graphRepresentation;
        //представя графа като матрица на съседство
}
```

Класът ShortestPath има следната структура:

```
class ShortestPath{

	Graph graph; //съдържа репрезентирането на графа
	int start; // начална точка
	int end; // крайна точка
	int [] nodesDistance; //масив, който съдържа текущото минимално разтояние между точките
	int [] parentsDistance; //масив, който съдържа индекса на предходния елемент с най-кратко разстояние до него
	boolean [] visited; //масив, който пази посетените възли на графа
	List<Pair> path; //списък, който конструира резултата

	private void initialize(){
		//инициализира масивите с подходящи стойности
	}

	private int getMinDistanceIndex(int [] nodesInstances, boolean [] visited){
		//намира индекса на елемента с най-малко разстояние
	}

	private List<Pair> dijkstra(){
		//реализира алгоритъма
	}

	private void updateNeighboursPath(int currentMinDistanceIndex, int size){
		//променя стойностите на съседите на текущия минимален елемент
	}

	private List<Integer> getShortestPath(){
```

```
        //пуска алгоритъма на Дийксра и обработва резултата от
    него
    }

}
```

3. Описание на алгоритъма.

function dijkstra(Graph, start, end):

initialize **nodesDistance** values to INFINITY(max value of Integer)
initialize **visited** values to FALSE
initialize **nodesParents** to UNDEFINED (-1)

set **nodesDistance[start]** to 0
set **nodesParents[start]** to 0

set **visited[start]** to true

while **visited[end]** is not true:

currentMinDistanceIndex = get the index of the closest node

set **visited[currentMinDistanceIndex]** to true

//update all unvisited neighbours of the current minimum element:

for each **neighbour** of **currentMinDistanceIndex**:

distanceSum = nodesDistances[**currentMinDistanceIndex**] + graph[**currentMinDistanceIndex][neighbour]**

if **distanceSum < nodesDistance[currentMinDistanceIndex]**

set **nodesDistance[currentMinDistanceIndex] = distanceSum**

return **nodesDistance[], nodesParents[]**