Universität Stuttgart

Institut für Visualisierung und Interaktive Systeme

Assignment 4

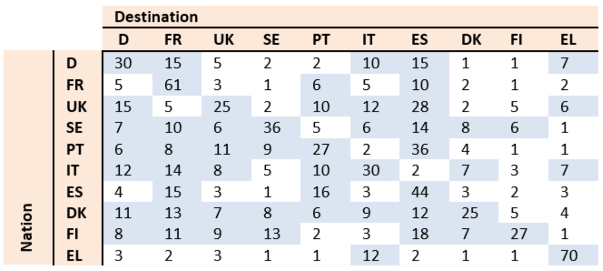
Information Visualization & Visual Analytics (WS 2019/20)

Due: Monday, 18.11.2019, 11:59 AM **Discussion:** Wednesday, 21.11.2019

Please solve the assignment in **groups of up to three (3) students**. Choose one student, who uploads your solution on the assignments page in ILIAS as PDF (for theoretical submissions) or ZIP (for practical submissions $\boxed{\text{Impl}}$). The submitted files should follow the naming scheme `yourlastname1_yourlastname2_yourlastname3` with respective file extension, of course. Make sure that you create your team before uploading the solution.

Task 1 Graph Representation *[Points: 15]*

Graphs are widely used to model relations between a set of objects. Sometimes, as in this exercise, the relations are weighted and directed. The graph described by the table below is complete, i.e., every object is related to all other objects.



		Destination									
		D	FR	UK	SE	PT	IT	ES	DK	FI	EL
Nation	D	30	15	5	2	2	10	15	1	1	7
	FR	5	61	3	1	6	5	10	2	1	2
	UK	15	5	25	2	10	12	28	2	5	6
	SE	7	10	6	36	5	6	14	8	6	1
	PT	6	8	11	9	27	2	36	4	1	1
	IT	12	14	8	5	10	30	2	7	3	7
	ES	4	15	3	1	16	3	44	3	2	3
	DK	11	13	7	8	6	9	12	25	5	4
	FI	8	11	9	13	2	3	18	7	27	1
	EL	3	2	3	1	1	12	2	1	1	70

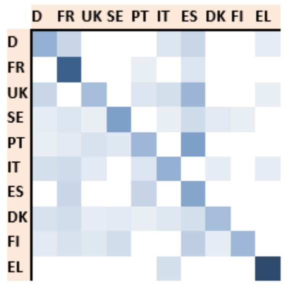
The table shows preferred holiday destinations (columns) by nations (rows). The fictitious numbers represent the percentage of survey participants, who prefer the destination accordingly (Note: not all states of the survey are included in the table and therefore the percentages per row do not sum up to 100 percent). The percentage of participants that prefer to stay within their state for vacation can be found along the main diagonal. Note that the matrix is not symmetric.

(a) *(4 points)* Design (sketch) and describe a visualization for the weighted directed graph described by the table above! Your visualization should not be based on a matrix or a bar chart. You may use a threshold and include only relations with a weight bigger than 5 (marked in blue). Map the weights and direction within your visualization.

(b) *(2 points)* How would you visualize additional information of the states within your solution? Please extend your visualization, so that it visually provides information about the state's total population and the average temperature. This information must not be encoded as a label/text! (You can use fictitious values.)

(c) *(4 points)* Compare your visualization (from (a)) with the matrix visualization below and name two benefits and two drawbacks. Describe in how far both techniques are suitable to visualize the complete graph (without applying the threshold).

(d) *(5 points)* **Bonus Assignment** $\boxed{\text{Impl}}$ Implement your sketch visualization in a language of your choice, and load both provided matrices in it. Discuss the limitations of your implementation in terms of scalability.



Task 2 $\boxed{\text{Impl}}$ Force-directed Graph Layouts *[Points: 10]*

Graphs are often visualized using node-link diagrams. The position of the graph nodes is usually determined by a layout algorithm. Force-based layout algorithms are based on attracting and repelling forces. Generally nodes repel from each other and additionally related nodes are attracted to each other (see page 36ff. of the lecture slides "Graphs and Networks").

In this exercise you have to implement a force-based layout algorithm as proposed by Fruchterman and Reingold [1]. You can find a pseudo-code immplementation of the algorithm in the lecture notes. Please note that the FdlHelper class provides most of the pseudo-code's functions (e.g., the cooldown function). Also, make sure to use the dimensions provided in the helper class for this assignment.

(a) *(3 points)* Draw the initial layout by calling the *renderCanvas(...)* method.

(b) *(5 points)* Draw the result after each iteration of the algorithm. You can use the FdlHelper class (*getGraphGeometry(...)*) to get the graph's geometry. Use of the method *Thread.sleep(ms)* to make the intermediate results visible.

(c) *(2 points)* Include the results (i.e. the initial graph, a couple of the intermediate results and the final result) in the written submission (the PDF document) of the assignment.

The data provided with this assignment contains:

- a Graph and a GraphNode class,
- a helper class that provides the parameters iterations, the area that should be used (L and W in the lecture's pseudo code), the cool down function cool(t), and the functions fr(x) and fa(x)
- the DataProvider class that provides a graph.

Data structure of this assignment:

DataProvider

getGraph(): Graph – returns the graph that has to be laid out

Graph

getNodes(): List ¡ GraphNode ¿ – returns a list of all nodes contained within the graph

GraphNode

getAdjacentNodes(): List ¡ GraphNode ¿ – returns a list of all nodes that are connected to this node.
Note: the graph does not contain explicit edges! Get neighboring nodes with this method.
getNodeID(): int – returns an ID unique to this node
getPosition(): Vector2D – returns the node's position.
setPosition(Vector2D position): Vector2D – sets the node's position.
getDisplacement(): Vector2D – returns the node's displacement.
setDisplacement(Vector2D displacement): Vector2D – sets the node's displacement.

[1] Fruchterman, Thomas MJ, and Edward M. Reingold. "Graph drawing by force-directed placement."

DataProvider

int iterations – represents the number of iterations that the algorithm should run
double width – represents the width of the space made available to the algorithm
double height – represents the height of the space made available to the algorithm
cool(int t): double – the cooling function used to reduce the force applied to the nodes
fr(double x): double – the repulsive force function used in the Fruchterman-Reingold algorithm
fa(double x): double – the attractive force function used in the Fruchterman-Reingold algorithm
getGraphGeometry(Graph g): List¡AbstractGeometry¿ – helper method to visualize the graph

Task 3 Presentation of results *[Points: 2]*

(a) *(2 points)* You may receive up to two bonus points for presenting either of the two exercises during the tutorial.