Second-Year Project
Programming Part

Project description

For the programming part of this course, you will write and submit a separate report. Your report should include the following.

- An introduction, in which you specify the problem and explain the outline of your report (max 500 words).
- Problem 10 (describe the data structures).
- Problem 11 (shortly describe the implementation, focus on writing what is asked in the problem).
- Problem 12 (shortly describe the implementation, focus on writing what is asked in the problem).
- Problem 13 (discuss the investigation you made).
- Problem 14 (describe which extensions you made and discuss the results).
- A discussion and conclusion, in which you reflect on your program and report, discuss its limitations and make suggestions for improvements.

This second report should consist of maximum 2000 words. Do not add appendices or your code! You can describe your implementation in high level pseudocode if you wish, but do not include actual source code. Name the report according to the following example of the team consisting of Jan Klaassen and Yolanda Xantippe:

KlaassenXantippeReport2.pdf

Please upload the source code of your Java program separately. Create a .zip file containing all the relevant .java files. Name it according to the following example of the team consisting of Jan Klaassen and Yolanda Xantippe:

KlaassenXantippeCode.zip

Upload your report (a single .pdf file) and source code (a single .zip file) on the correct upload points on the course page. The deadline of the report and source code of part 2 is Friday, January 27, at 23.59 (CET).

Introduction of your report (*10 points*)

Your report should start with a introduction in which you specify the problem (see below) and explain the outline of your report.

Problem description

Now that you are familiar with the process of iterated elimination of strictly dominated strategies, let us turn our attention to computer aided solutions.

Consider Problem 1 again from part 1. Let us formulate this in the following, general way. In a certain region there are n firms and m cities. Every city i has $size(i)$ inhabitants, and the distance between every two cities i and i' is $dist(i, i')$. The firms compete with each other for the market of a certain product in this region. Every firm j must choose a location $loc(j)$ for its store from the different possible locations.

For now, we will assume that the set of locations coincides with the set of cities. That is, every firm can choose a location for its store from the different cities. In general, there might be a set of k possible locations, distinct from the set of cities, and $dist(a, b)$ is a function that represents the distance between a and b, where a and b are a city or a location.

We assume that the price for the product is fixed, and that every inhabitant will always buy one item of the product from the nearest firm. If two or more firms are located in the same location, we assume that they will share the market equally among them. The objective of every firm is to sell as many items of the product as possible.

One of the concepts that you will use to analyse this game is the iterated elimination of strictly dominated choices. You applied this already in Problem 1 for a small instance, which could still be executed by hand.

However, an application to a real life case might quickly become cumbersome in a situation with more possible locations. Moreover, some additional side constraints could necessitate the use of a variant of the procedure, tailored to that situation. We will continue our investigations using automated computing in a program that you will write.

Problem 10: Data structures (*20 points*)

A *graph* $G = (V, E)$ consists of a set of *vertices* (or nodes) V and a set of undirected *edges* $E \subseteq V \times V$. When two vertices $u, v \in V$ are connected by an edge $e = \{u, v\}$, we say u and v are *neighbours*. Such a graph structure

can be used to model the set of cities in a certain region, where two cities are connected by an edge if they are "close".

Take a look at the following example. Consider the cities of Maastricht, Heerlen, Sittard and Roermond. Assume for now that the possible locations of the firms correspond to this set of cities. You could look up the number of inhabitants and the distances between these four cities. Suppose there are two firms. Then we could model the cities of the question above by a graph consisting of four vertices representing the cities of Maastricht, Heerlen, Sittard and Roermond, and connect two cities if and only if the straight distance between them is at most, say, 30 kilometers. Notice that if we vary this distance parameter, the graph will change: For higher values, more cities will be connected by a direct edge, and for lower values, there will be less edges in the graph.

Task. Write a computer program that can read in a .txt input file. A correctly formatted input file specifies the number of cities on the first line. Then for each of these cities, it specifies its name, the number of inhabitants, its longitude coordinate and its latitude coordinate. The input file for the example above is provided on the module on the course page.

This program should then construct a graph of the region. It should create a node for each city based on its longitude and latitude coordinates and connect two cities by an edge if and only if their direct distance is at most some threshold (whose value you decide).

Create an input file from a region somewhere in the world with roughly 50 cities (it can be a bit more or a bit less). Choosing your favourite region will personalize this project – we hope every team chooses a different region.

(In this data structure, you are allowed to work under the assumption that the set of cities and the set of locations coincide. A possible extension in Problem 14 is to relax this assumption and to have separate sets of cities and locations. If you decide to implement this extension, your data structure should reflect this. See Problem 14.)

Assessment. You can obtain a maximum of 15 points for the most efficient implementation of your data structures, and you can obtain a maximum of 5 points for the part of the report that describes this on a high level.

Problem 11: Iterative elimination (*20 points*)

Task. Take the data structure that you implemented in Problem 10. Add a method to your program that takes a given configuration of cities and two firms as input, and finds the locations that survive the iterated elimination of strictly dominated choices. Note that this method should be able to work for any configuration of cities and is independent of the edge structure of your graph.

Test this program on the cities from the region you chose in the previous question, where you can use the number of inhabitants of the cities to represent the demand. Make a visual representation of the results and try go give some intuition why some cities survive and others were eliminated. Is it true that if a city is strictly dominated by another city, then it will also be strictly dominated by a neighbouring city?

Assessment. You can obtain a maximum of 15 points for the most efficient implementation of this procedure, and you can obtain a maximum of 5 points for the part of the report that describes this on a high level.

Problem 12: Local search (*20 points*)

This iterated elimination of strictly dominated strategies is not the only way for each firm to decide on a location. Another strategy would be to look at the results of local search dynamics. Here, every firm has an initial (hypothetical) starting location. Then, iteratively, every firm considers the current location of the other firm and computes whether it would be beneficial to move to a neighbouring location. If so, it moves to the neighbouring location that gives this player the highest utility. Then the next firm might reconsider its current position, based on the location of all other firms. This continues until no firm has an incentive to unilaterally deviate from its current location (or until another stopping criterium is reached).

Task. Implement such local search dynamics to try to find an equilibrium (if the iterative procedure stops). Compare the results with your results for the previous question.

Assessment. You can obtain a maximum of 15 points for the most efficient implementation of this procedure, and you can obtain a maximum of 5 points for the part of the report that describes this on a high level.

Problem 13: Investigating the local search procedure (*5 points*)

Task. Do you obtain different results with different starting positions? Or if you take different values for the parameters (e.g. the parameter that decides the maximum distance between two cities to be connected by an edge)? Or if you use different stopping criteria?

Make an investigation of the influence of these factors on the results, and report on this.

Assessment. You can obtain a maximum of 5 points for the part of the report that discusses this investigation.

Problem 14: Extending your program (*15 points*)

Task. Add some whistles and bells to your program to bring it closer to a real life case scenario. Here are some suggestions, but feel free to incorporate other nice extensions in your program that make it more practical.

- The first suggested extension was already mentioned before: Relax the assumption that the set of cities and the set of locations coincide. That is, besides the set of cities, there is a (possibly overlapping) set of locations that the firms could choose. Firms do not select an element from the set of n cities any more, but an element from a set of k locations. In this extension, you might want to change the input file and/or the distance matrix you created.

  If you implement this suggestion, (1) shortly describe how your program changed, (2) visualize the possible set of cities and the possible set of locations in your report, and (3) shortly discuss the effects of this change on your results.

- Another suggested extension is the situation in which customers have a maximum distance they are willing to travel, and that firms only attract customers if they are located within this distance bound from the city where the customers live. If two firms both locate in Maastricht as their sole location in the Netherlands for example, customers from Groningen will simply be lost and they will not travel all the way to Maastricht.

If you implement this suggestion, (1) shortly describe how your program changed, (2) discuss the effects of this change on your results, and (3) shortly discuss the influence of the value of this bound on the customers' willingness to travel on your results.

- You could also think of a model where firms include prices for their product or service. Every customer now has a budget of how much he or she is willing to spend on the sum of the distance and the price of the product or service. Every customer will then go to the firm which has the lowest cost for them in terms of distance plus price, if there is a firm within their budget.

  If you implement this suggestion, (1) shortly describe how your program changed, (2) discuss the effects of this change on your results, and (3) shortly discuss the influence of the value of this budget on your results.

  If you want to take this suggested extension even further, perhaps you can compute in your local search dynamics not only a possible new location, but also a possible new price of the firm, where they want to maximize their number of customers times the price they offer. In this case, describe the design choices you made in this combined local search procedure and its effect on the results.

Again, feel free to incorporate other nice extensions in your program that make it more practical.

Assessment. You can obtain a maximum of 10 points for all extensions that you incorporate, and you can obtain a maximum of 5 points for the part of the report that describes these extensions.

Conclusion of your report (*10 points*)

Your report should finish with a conclusion and discussion, in which you reflect on your program and report, discuss its limitations and make suggestions for improvements.