Determining optimal route using fuzzy logic and Dijkstra's algorithm

Julia Iskierka, Szymon Lipiec
Faculty of Applied Mathematics
Silesian University of Technology
Kaszubska 23, Gliwice, 44-100, Poland
e-mails: jul.iskierka@gmail.com, lipieczsp6@gmail.com

***Abstract*—How to find optimal route between two cities of Silesian voivodeship taking into consideration four factors - road's distance, condition, traffic level and maximum speed? A combination of fuzzy logic, graphs theory and Dijkstra's algorithm is an answer for this quation. Taking all this components, a simple system solving this problem was presented in this paper. A weighted graph with edges' weights defined by fuzzy logic was applied in Dijkstra's algorithm and a route between two given cities was found as a result. The proposed solution and conducted experiments are described and discussed in this paper.**

I. Introduction

In XII century technical development and geopolitical situation caused sudden increase of travel possibilities and therefore showed up new problems connected with transport. People could not memorize all the roads and using regular paper map costed valuable time, not necessarily giving the best of possible ways. Furthermore, there was no possibility of measuring other factors than distance, like traffic level or road's condition, which are also very important. So people began to look for mathematical models to solve cumulative problems. New way of describing the world was invented - fuzzy sets [1].

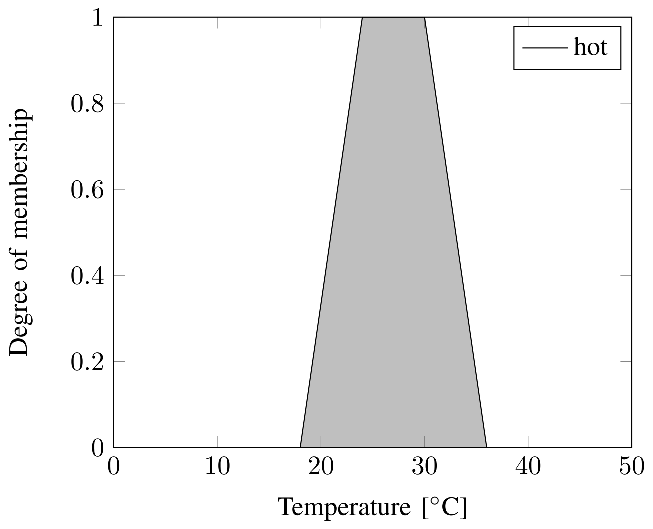
Fuzzy sets are an important element of today's computer science, which find a wide range of applications in practical problems. To this day, mathematical aspects are being expanded and improved, what can be seen on the example of the presented works around the world. In [2], an efficient fuzzy logic system with triangular type-2 sets was presented. Again in [3], a fuzzy controler was used for purposes of making decisions in the object recognition system. Fuzzy sets were used for inference [4]. Moreover, existing graph theory was used to present maps what was described in [5] and algorithms determining the shortest path (for example Dijkstra's algorithm) were created [6].

We connected mentioned methods in one system, which finds the best from available roads, given various factors.

II. Mathematical part

A. Fuzzy sets

a) Usage: Fuzzy sets are used in situations when we can not describe things in true or false terms. If some people are asked to give a specific temperature when it is hot, the answers will be very various and they will rather give a scope than a number. This situation is a perfect example of when to use fuzzy set, as shown below:



At the example fuzzy set above we can see that for values 18°C - 24°C it is hot in some extent defined by the *membership function*, then from 24°C to 30°C it is definitely hot and finally, again, definition of hotness is not so clear. Such functions are based on expert knowledge - experience, statistics or other sources. It depends on modeled system.

b) Definition: Fuzzy set A in set X is a pair:

$$A = \{(x), \mu_A(x) | x \in X) \tag{1}$$

where $\mu_A : X \to [0, 1]$ is a membership function. It is often denoted by (proposed by Lotfi Zadeh):

- for finite set X:

$$A = \sum_{i=1}^{n} \mu_A(x)/x, \tag{2}$$

where / is not division mark and sum does not mean summing, but the fuzzy set being built from elements x_i of finite set X and being described by membership function $\mu_A(x)$;

- for infinite set X:

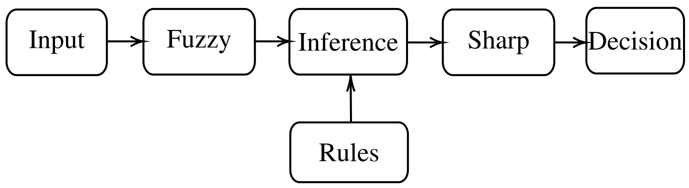
$$A = \int_{x \in X} \mu_A(x)/x, \tag{3}$$

  where / is not division mark and sum does not mean summing, but the fuzzy set being built from elements x_i of infinite set X and being described by membership function $\mu_A(x)$.

Membership functions have many shapes, but the most used ones are trapezoidal, triangular, Gaussian and sigmoid.

B. Mamdani's fuzzy inference system

With help of fuzzy sets it is possible to make a decision making system. For example, we wonder if it is a good idea to go running given the time of the day and the temperature. Using suitable mathematical method, this decision can be easily made. One of those methods is Mamdani's fuzzy inference system. It's schematic way of working is shown below:



Scheme of how Mamdani's fuzzy inference system works

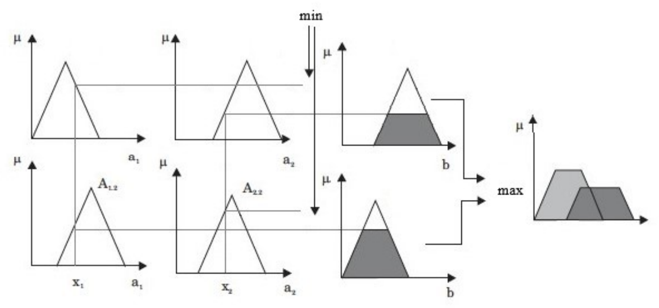
Input: Input vector with numerical description of parameters, on the basis of which we want to make a decision (like time and temperature).

Fuzzy: Calculating the outcome of proper membership function for each element of input vector.

Rules: Set of rules made from linguistic expressions of natural language, which are later transformed into numeric values needed in the decision making process. They are built in specific way, as shown below:

*If time is **late** and temperature is **high**, then probability of running is **high***

Inference: Having collected the data from set of rules, we can carry out inference. The easiest way to explain that is to show it on graph.



Inference visualization

We have two parameters having their own membership functions, defining their states (like hot, early). We convert crisp input values into fuzzy ones and take minimum of them for logical operator *or* between the parameters and maximum for *and*. Later, we aggregate the results of all rules by taking their maximum.

Defuzzification: Sharpening the result of inferencing (fuzzy set) into numeric value. One of defuzzification methods is centre of gravity method. Outcome is calculated from formula:

- for finite sets

$$output = \frac{\sum_{i=1}^{n} x_i \mu(x_i)}{\sum_{i=1}^{n} \mu(x_i)}, \tag{4}$$

- for infinite sets

$$output = \frac{\int_0^X x_i \mu(x_i) dx}{\int_0^X \mu(x_i) dx}. \tag{5}$$

C. Graph theory

a) Graph definition: Graph is a mathematical structure used for showing relations between objects. In a simplified way, it is a collection of vertices (or nodes) and connections between them, called edges. There are specific types of graphs i.a.:

- weighted graph - edges have weights
- undirected graph - edges do not have orientations

b) Adjacency matrix: It is a square matrix used to represent graph. Value of element a_{ij} indicate if there is a connection (edge) between vertices i and j.

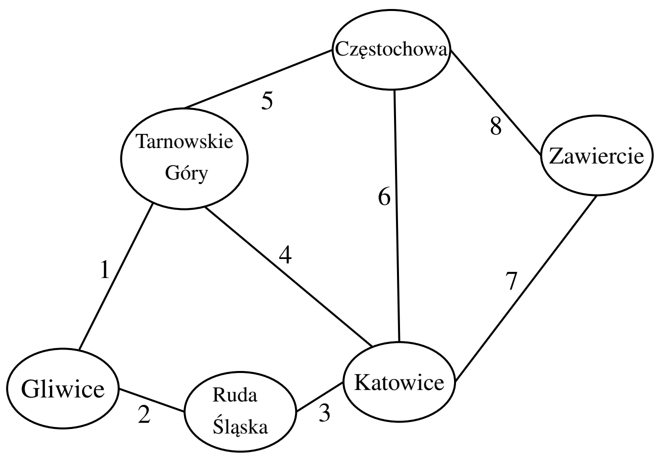
c) Dijkstra's algorithm: Algorithm used for finding the most optimal route between vertices of a graph. It is a greedy algorithm, so it makes locally optimal choices, which means that the overall result may not always be the best one.
Algorithm step by step:

1) Mark all vertices unvisited and make a set of them. Label the starting vertice as current.
2) For each vertice assign a distance - zero for the one you start from, infinity for the rest.
3) Calculate distance from starting point for all unvisited neighbours of current vertice. Compare the new value with the old one and save smaller.
4) Remove the current vertice from the unvisited set (it will not be checked again).
5) If all vertices have been visited or the smallest distance among vertices in unvisited set is infinity, then stop.
6) Otherwise, set the vertice with shortest distance as current and go to step 3.

III. Proposed system

A. *Mathematical model*

Weighted, undirected graph was made, where cities of Silesian voivodeship were vertices and roads between them were edges. The weights consisted of length, condition, level of traffic and maximum speed. To save the data in memory a slightly modified adjacency matrix was used - values inside it kept not only the information about existence of the road, but also it's weight.



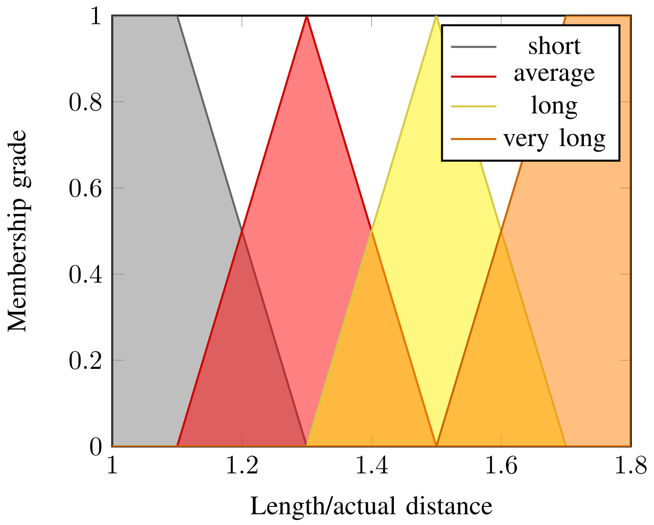
Graphical representation of used graph

ID	Length	Condition	Traffic level	Max speed
1	27,4	0,8	0,1	100
2	15,8	0,9	0,2	140
3	13,9	0,9	0,3	140
4	27,2	0,7	0,6	60
5	53,7	0,5	0,4	80
6	74,6	0,6	0,7	100
7	45,3	0,7	0,3	80
8	56,9	0,5	0,1	60

Table I: *Table of roads' weights*

Fuzzy sets were modelled for every parameter of the road's weight.

Membership function of length to actual distance ratio

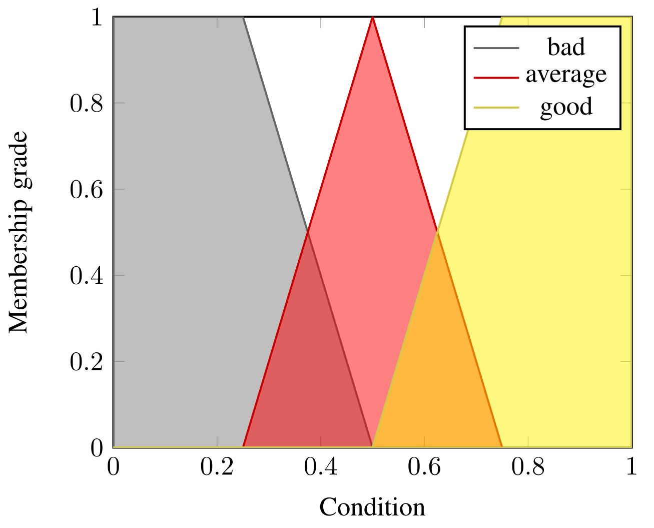


Length of the road to actual distance ratio was used to check how much "unnecessary" track has to be driven. Choosing only length would make no sense, because it would always return the same values, no matter if the road would lead in straight line or be a winding one (so a longer one). For example, if there would be two roads from city A to B, one longer that another, but not long enough to qualify to different fuzzy set, the result might be the same for both. The method using ratio takes it into consideration.

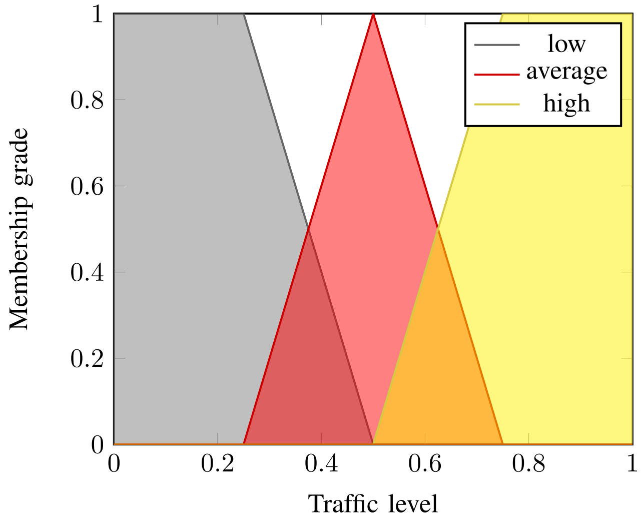
The actual distance between cities was calculated using geographical coordinates and formula:

$$\sqrt{(x_2 - x_1)^2 + (\cos\frac{x_1 \cdot \pi}{180} \cdot (y_2 - y_1))^2} \cdot \frac{40075,704}{360} \qquad (6)$$

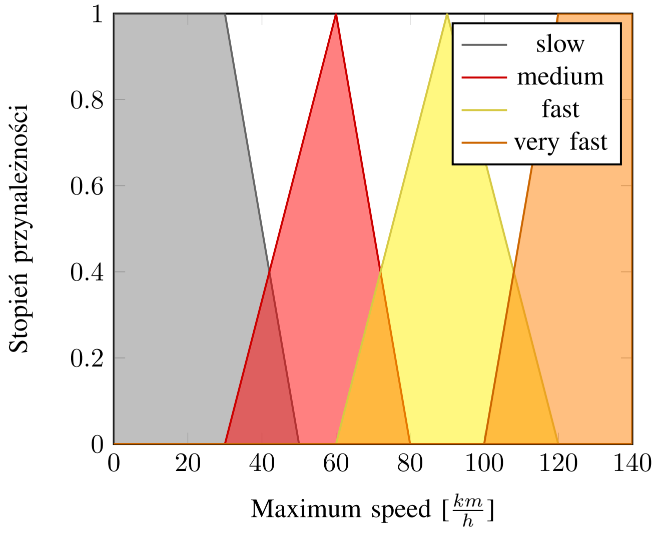
Membership function of road's condition



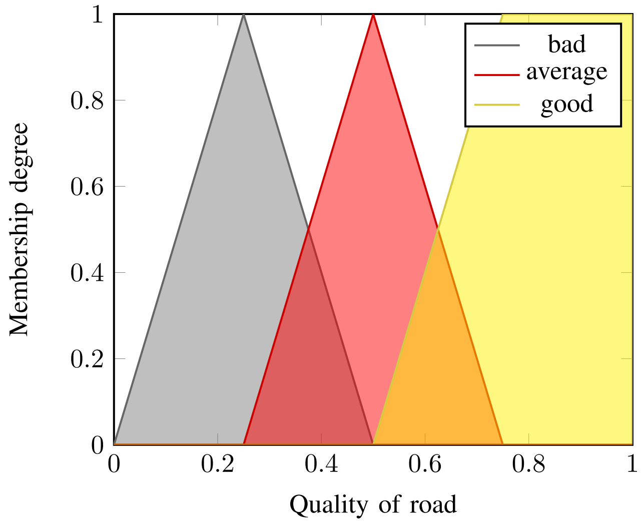
Membership function of traffic level



Membership function of maximum speed



Output membership function



Set of rules was implemented. They were divided into groups for each linguistic expression in output set (bad, average, good), for example:

*If length to actual distance ratio is **short** and condition is **good** and traffic level is **low** and max speed is **fast**, then quality of road is **good***

*If length to actual distance ratio is **short** and condition is **average** and traffic level is **high** and max speed is **fast**, then quality of road is **average***

*If length to actual distance ratio is **very long** and condition is **bad** and traffic level is **high** and max speed is **slow**, then quality of road is **bad***

Having been applied, Mamdani's inference system evaluated weight of every road in adjacency matrix. Dijkstra's algorithm used this modified matrix to determine optimal road between given cities.

B. Pseudocode

```
Data: City database
Data: Road database
Input : Adjacency matrix
Output: Fuzzy adjacency matrix
for $i \leftarrow 1$ to table dimension do
    for $j \leftarrow 1$ to table dimension do
        if $Table[i, j] == -1$ then
            $Fuzzy[i, j] = 0$
        else
            $Fuzzy[i, j] = 1 -$ Valuation(Table[i,j])
        end
    end
end
```

```
foreach verticle v in V[G] do
    d[v] := $\infty$
    parent[v] := unknown
end
d[s] := 0
Q := V
while Q not empty do
    u := TakeOffMin(Q)
    foreach verticle v - neighbour u do
        if d[v]>d[u]+w(u,v) then
            d[v]:=d[u]+w(u,v)
            previous[v] := u
        else
            Continue
        end
    end
end
Print(Shortest path is d[v])
```

IV. Tests

We took two cities - Gliwice and Zawiercie to check *Dijkstra's algorithm*. As shown on Table II, each route's weight was determined using fuzzy logic.

ID	City A	City B	Weight
1	Gliwice	Tarnowskie Góry	0,366
2	Gliwice	Ruda Śląska	0,250
3	RudaŚląska	Katowice	0,312
4	Katowice	Tarnowskie Góry	0,500
5	Tarnowskie Góry	Częstochowa	0,500
6	Częstochowa	Katowice	0,500
7	Katowice	Zawiercie	0,429
8	Zawiercie	Częstochowa	0,500

Table II: Roads' weights

On this small example it is easy to calculate without any algorithm, that the finest route from Gliwice to Zawiercie leads through Ruda Śląska and Katowice. Outcome of Dijsktra's algorithm matches with this calculation.

ID	City A	City B	Weight
1	Gliwice	Tarnowskie Góry	0,366
2	Gliwice	Ruda Śląska	0,750
3	RudaŚląska	Katowice	0,672
4	Katowice	Tarnowskie Góry	0,500
5	Tarnowskie Góry	Częstochowa	0,429
6	Częstochowa	Katowice	0,500
7	Katowice	Zawiercie	0,429
8	Zawiercie	Częstochowa	0,500

Table III: New roads' weights

Then we changed weights of roads what resulted in new adjacency matrix (shown in Table III). This time, the best road leads through Tarnowskie Góry and Częstochowa. Again Dijkstra's algorithm returned the same results as expected.

V. Conclusions

Connecting described mathematical models - fuzzy sets, graphs theory and Dijkstra's algorithm - can result in properly working system for determining optimal route between two points at the map. In this project data regarding road status was set statically, but if some sensors or processed satellite's images were used to provide the data and graph was expanded, it could be utilized as navigating system. However, it's efficiency might not be the best, as Dijkstra's algorithm looks through all vertices, so it could be replaced by different route finding algorithm.

Defining fuzzy sets also needs a closer look. As they are created on basis of experts knowledge, it is not easy to check if we made them properly. Some research should be made to have a foundation to work on and determine them the finest way.

References

[1] L. A. Zadeh, "Fuzzy logic," *Computer*, vol. 21, no. 4, pp. 83–93, 1988.

[2] J. T. Starczewski, "Efficient triangular type-2 fuzzy logic systems," *International journal of approximate reasoning*, vol. 50, no. 5, pp. 799–811, 2009.

[3] M. Woźniak and D. Połap, "Object detection and recognition via clustered features," *Neurocomputing*, vol. 320, pp. 76–84, 2018.

[4] C.-C. Lee, "Fuzzy logic in control systems: fuzzy logic controller. ii," *IEEE Transactions on systems, man, and cybernetics*, vol. 20, no. 2, pp. 419–435, 1990.

[5] D. B. West *et al.*, *Introduction to graph theory*. Prentice hall Upper Saddle River, NJ, 1996, vol. 2.

[6] S. E. Dreyfus, "An appraisal of some shortest-path algorithms," *Operations research*, vol. 17, no. 3, pp. 395–412, 1969.

[7] O. Cordón, "A historical review of evolutionary learning methods for mamdani-type fuzzy rule-based systems: Designing interpretable genetic fuzzy systems," *Int. J. Approx. Reasoning*, vol. 52, no. 6, pp. 894–913, Sep. 2011. [Online]. Available: http://dx.doi.org/10.1016/j.ijar.2011.03.004

[8] B. Kosko and M. Toms, *Fuzzy thinking: The new science of fuzzy logic*. Hyperion New York, 1993.

[9] A. Fornaia, C. Napoli, G. Pappalardo, and E. Tramontana, "Enhancing city transportation services using cloud support," in *International Conference on Information and Software Technologies*. Springer, 2016, pp. 695–708.

[10] A. Fornaia, C. Napoli, and E. Tramontana, “Cloud services for on-demand vehicles management,” *Information Technology and Control*, vol. 46, no. 4, pp. 484–498, 2017.