From Fuzzy to Mobile Fuzzy

Victor L. Timchenko[1], Yuriy P. Kondratenko[2],
Olga Kosheleva[3], and Vladik Kreinovich[4]

[1]Admiral Makarov National University of Shipbuilding,
9 Heroes of Ukraine Avenue
Mykolaiv 054025, Ukraine, vl.timchenko58@gmail.com

[2]Petro Mohyla Black Sea National University
10, 68th Desantnykiv Str.
Mykolaiv 054003, Ukraine, and
Institute of Artificial Intelligence Problems
Ministry of Education and Science (MES) of Ukraine and
National Academy of Sciences (NAS) of Ukraine
Kyiv 01001, Ukraine
y_kondrat2002@yahoo.com

[3]Department of Teacher Education
[4]Department of Computer Science
University of Texas at El Paso
500 W. University
El Paso, Texas 79968, USA
[3]olgak@utep.edu, [4]vladik@utep.edu

Abstract

The main limitation of mobile computing in comparison with regular computing is the need to make sure that the battery last as long as possible — and thus, the number of computational steps should be as small as possible. In this paper, we analyze how this affects fuzzy computations. We show that the need for fastest computations leads to triangular membership functions and simplest "and"- and "or"-operations: min and max. It also leads to the need to limit ourselves to a few-bit description of fuzzy degrees — which leads to 3-bit descriptions similar to optical implementation of fuzzy computing.

1 Formulation of the Problem

Need for fuzzy techniques. In many practical situations we rely on experts to make good decisions. For example, in medicine, in spite of numerous successes of automatic systems, we still rely on human doctors to make decisions.

In each application area, some experts are very good, others are not that experienced. It would be great if everyone could be served by the best experts, but there are usually only a few of them, and they cannot help everyone. So, a natural idea is to incorporate the knowledge of top experts into a computer system that will help other experts.

Experts are usually willing to share their knowledge, but the problem is that they usually describe their knowledge in terms of imprecise ("fuzzy") words from natural language, and it is not easy to translate these words into computer-understandable terms.

For example, in the US, the vast majority of people knows how to drive. However, when you ask a person what to do if you are driving on a freeway at a speed of 65 miles per hour and a car 30 feet in front of you slows down to 60 – a typical answer is "brake a little bit". The problem is that computers do not understand natural-language words like "a little bit", they need to with what exactly force and for how many milliseconds to apply the brakes.

Techniques for translating from "fuzzy" natural language into precise terms are called *fuzzy technqiues*; see, e.g., [1, 2, 3, 5, 6, 11]. These techniques have been successfully applied in many areas.

Specifics of mobile computing and the resulting general problem. The main limitation of mobile computing in comparison with regular computing is the need to make sure that the battery last as long as possible — and thus, the number of computational steps should be as small as possible.

What we do in this paper. In this paper, we analyze how the mobile-computing-related limitation affects fuzzy computations.

Structure of the paper. The structure of this paper is as follows. In Section 2, we remind the readers of the main computational steps related to fuzzy computations. In a short Section 3, we remind the readers that there are two ways to decreases the number of computational steps: to decrease the number of arithmetic operations and to decrease the number of bits in the representation of the corresponding numbers. In Sections 4 and 5, we show how we can do this for fuzzy computations. Conclusions form Section 6.

2 Fuzzy Techniques: a Brief Reminder

General idea. How can we describe terms like "little" in precise terms? For quantities like "positive", their meaning is very clear:

- if a number x is negative or zero, i.e., if $x \le 0$, then this number is positive; but

- if a number x is larger than 9, then this number is positive.

As we go from small negative numbers to 0, our opinion about the statement "x is positive" drastically changes from "false" to "true". In contrast, for fuzzy properties like "x is small", there is no such abrupt transition:

- values x close to 0 are absolutely small,
- values x which are much larger than 0 are absolutely not small, and
- intermediate values are small *to some extent*.

In a computer, "true" is usually represented as 1, and "false"as 0. Thus, it is reasonable to represent degrees of certainty intermediate between "absolutely true" and "absolutely false" by numbers intermediate between 0 and 1.

The notion of a membership function – or, equivalently, a fuzzy set. In line with the above general idea, each natural-language property like "small" is described by assigning, to each possible value x of the corresponding quantity, a degree $m(x) \in [0,1]$ to which this value x satisfies the given property – e.g., the degree to which x is small.

Informally, this function $m(x)$ describes to what extent the value x belongs to the set of all small numbers. Because of this informal description, the function $m(x)$ is called a *membership function*, or, alternatively, a *fuzzy set*.

Fuzzy logic: general description. In practice, expert-provided rules often use logical connectives like "and", "or", and "or". For example, in the car case, the condition was that the speed is 65, *and* that the other car is at 30 ft, *and* this other car slows down to 60. Medical recommendations are also full of such rules.

These logical connectives are easy to apply if we are dealing with precise statements. In this case, the truth values of two statements A and B uniquely determine the truth values of the corresponding logical combinations $A \,\&\, B$, $A \vee B$, and $\neg A$.

The situation is different in the fuzzy case. If we know that the statement A holds with degree of confidence 0.8, and the statement B holds with degree of confidence 0.9, what is the degree of confidence in $A \,\&\, B$?

Fuzzy "and"-operations. In the ideal world, we should ask the same expert – whom we asked to gauge his/her degree of certainty in two statements A and B – to also gauge his/her degree of certainty in the combined statement $A \,\&\, B$. We can do it for one or two combined statements, but for n basic statements, there are exponentially many such combined statements, and it is not possible to ask the expert about all of them.

Since we cannot always elicit the degree of confidence in a combined statement $A \,\&\, B$ from the expert, we need to estimate this degree based on whatever information we have – i.e., based on our degrees of confidence $a = d(A)$ and $b = d(B)$ in statements A and B. In other words, we need to have a function that would input the degrees of certainty a and b in statements A and B and

return the estimate for the expert's degree of certainty in $A \,\&\, B$. Such a function is usually denoted by $f_{\&}(a, b)$ and is known as an *"and"-operation*, or, for historical reasons, a *t-norm*.

"And"-operations need to satisfy several natural conditions.

- First, for the cases when both degrees are 0s and 1s, they must coincide with the "and"-operation in the usual 2-valued ("yes"-"no") logic.
- Second, since usually, "A and B" means the same as "B and A", the estimates $f_{\&}(a, b)$ and $f_{\&}(b, a)$ for these two statements should coincide, i.e., we should have $f_{\&}(a, b) = f_{\&}(b, a)$. In mathematical terms, this means that the operation $f_{\&}(a, b)$ must be *commutative*.
- Similarly, since the statements "A and (B and C)" and "(A and B) and C" mean the same thing, the corresponding estimate $f_{\&}(a, f_{\&}(b, c))$ and $f_{\&}(f_{\&}(a, b), c)$ must also be equal. Thus, the operation $f_{\&}(a, b)$ must be *associative*.

Fuzzy "or"-operations. Similarly, we need a function $f_{\vee}(a, b)$ that estimates the degree of certainty in a combined statement $A \vee B$. This function is known as *"or"-operation*, or, for historical reason, a *t-conorm*.

"Or"-operations also need to satisfy several natural conditions.

- First, for the cases when both degrees are 0s and 1s, they must coincide with the "or"-operation in the usual 2-valued ("yes"-"no") logic.
- Second, since usually, "A or B" means the same as "B or A", the estimates $f_{\vee}(a, b)$ and $f_{\vee}(b, a)$ for these two statements should coincide, i.e., we should have $f_{\vee}(a, b) = f_{\vee}(b, a)$. In mathematical terms, this means that the operation $f_{\vee}(a, b)$ must be *commutative*.
- Similarly, since the statements "A or (B or C)" and "(A or B) or C" mean the same thing, the corresponding estimate $f_{\vee}(a, f_{\vee}(b, c))$ and $f_{\vee}(f_{\vee}(a, b), c)$ must also be equal. Thus, the operation $f_{\vee}(a, b)$ must be *associative*.

Fuzzy negation operations. To describe negation, we similarly need a *negation operation* $f_{\neg}(a)$. A natural condition is that for the cases when the input a is 0 or 1, the negation operation must coincide with the negation in the usual 2-valued ("yes"-"no") logic.

So what is fuzzy logic. The above logical operations with fuzzy degrees – as well as similar operations corresponding to implication and other logical connectives – form what is known as *fuzzy logic*.

3 How to Decrease the Number of Computational Steps: A General Reminder

Every computation consists of elementary steps. In a computer, the only directly hardware supported operations are arithmetic operations: addition,

subtraction, and multiplications. Computers also support min and max of two numbers. All other computations, whether we are computing the value of $\sin(x)$ or a solution of a complex partial differential equation, consist of a sequence of arithmetic operations.

For example, computation of $\sin(x)$ is usually done by computing the sum of several first terms in the corresponding Taylor series

$$\sin(x) \approx x - \frac{x^3}{3!} + \frac{x^5}{5!} + \ldots + (-1)^k \cdot \frac{x^{2k+1}}{(2k+1)!}.$$

Out of all elementary operations:

- the fastest are min and max,
- next fastest are addition and subtraction, and
- multiplication is the slowest: this makes sense, since multiplication requires several additions.

Each elementary operation consists of several bit operations. Each arithmetic operation, in its turn, consists of several bit operations. The more bits we use to represent each number, the more bit operations we need to preform a single arithmetic operation.

So how can we decrease energy consumption related to computations? Each bit operation requires some energy. Thus, to decrease energy consumption and to make mobile devices last longer, we need to decrease the overall number of bit operations. In line what we discussed, this means:

- decreasing the number of arithmetic operations – and selecting the fastest operations, and/or
- decreasing the number of bit operations needed for a single arithmetic operation, i.e., decreasing the number of bits used to represent each number.

In the following two sections, we will analyze how each of this ideas will affect fuzzy computations.

4 How to Minimize the Number of Arithmetic Operations in Fuzzy Computations

4.1 Let us start with fuzzy logical operations

How do we minimize energy consumption. To minimize energy consumption, we need to select an algorithm consisting of the smallest possible number of elementary operations, and these operations must be the fastest possible.

The smallest number of arithmetic operations that we can use is one, and the fastest operations are – as we have mentioned – min and max; next in speed are addition and subtraction.

Based on this, which "and"- and "or"-operations should we choose? The fastest possible arithmetic operations are min and max. Interestingly:

- min satisfies all the above-described properties of the "and"-operation, and
- max satisfies all the above-described properties of the "or"-operation.

Thus, in mobile implementation of fuzzy computations, it is reasonable to use $f_{\&}(a,b) = \min(a,b)$ and $f_{\vee}(a,b) = \max(a,b)$.

Comment. These operations are indeed successfully used in many applications of fuzzy techniques, where they lead to reasonable results.

Which negation operation should we choose? For negation operation, we cannot use min or max – neither of related operations $\min(a,c)$ or $\max(a,c)$ for some c coincides with the classical negation for both $a = 0$ and $a = 1$.

Next in speed are addition and multiplication. Among corresponding operations $a + c$, $a - c$, and $c - a$ the only one that coincides with the classical negation for both $a = 0$ and $a = 1$ is the operation $f_{\&}(a) = 1 - a$.

So this what we should use in mobile fuzzy computations. This operation is indeed actively and effectively used in fuzzy applications.

4.2 What about membership functions?

What we want. We want to describe a function $m(x)$ that assigns a number from the interval $[0,1]$ to any possible value x of the corresponding physical quantity.

What we need to take into account. It is important to take into account that the numerical value of a physical quantity depends on the choice of the measuring unit and on the choice of the starting point.

If we replace the original measuring unit with a new unit which is a times smaller, then all numerical value of the corresponding quantity get multiplied by a: $x \mapsto a \cdot x$. For example, if we replace meters with centimeters, all numerical values get multiplied by 100: e.g., 1.7 m becomes 170 cm.

Changing a measuring unit leads to transformation $x \mapsto a \cdot x$ for positive a. Sometimes, the change of sign also makes sense: for example, which direction of current shall we call positive and which negative is just a question of convenience. If we change the sign, then all numerical values change sign: $x \mapsto -x$.

Similarly, if we replace the original starting point with a new point which is b units lower, then this value b is added to all numerical value $x \mapsto x + b$. For example, if instead of the French Revolution calendar – that started in year 1789 – we use the usual calendar that started $b = 1789$ years earlier, then, e.g., French-calendar Year $x = 2$ becomes year $x + b = 1791$.

If we change both the measuring unit and the starting point, then we get a generic linear transformation $x \mapsto a \cdot x + b$.

How does this affect membership functions. We want to come up with a general expression for a membership function, an expression that would be useful

even if we change the measuring unit and/or the starting point for measuring the corresponding physical quantity. So, with each function $m(x)$ this family of functions should also contain all the functions of the type $m(a \cdot x + b)$ for different values a and b. We will say that this family is *invariant under re-scaling*.

Even if we start with the easiest-to-compute function $m(x) = x$ – that does not require any computations at all – we will still need to consider all linear functions $m(x) = a \cdot x + b$.

In general, computing a linear function requires two elementary operations: multiplication and addition. If we only use one arithmetic operation, i.e., use expressions of the type $x + b$, $2x$, $a \cdot x$, or $x \cdot x$, we do not get any re-scaling-invariant family. Thus, to get such a family, we need at least two arithmetic operations.

The fastest operations are addition and multiplication. However, if we use two additions or subtractions, we will get either $x + c$ or $c - x$ or $2x + c$ or $3x$ – so again, we do not get any re-scaling-invariant family.

Thus, we need at least one operation of next computation speed – i.e., multiplication. If we use one addition and one multiplication, then we get either $x^2 + c$ – which is not invariant – or $a \cdot x + b$.

Thus, locally, the simplest-to-compute membership functions should be linear.

Resulting recommendation. So, we should use piecewise-linear membership functions in fuzzy mobile computing.

Comment. Piecewise-linear membership functions – with triangular and trapezoid shape – are indeed used frequently and effectively in applications of fuzzy techniques.

5 How to Minimize the Number of Bits Per Number in Fuzzy Computations

What numbers are currently used to represent degrees of confidence. As we have mentioned, a usual way to represent a fuzzy degree is by using numbers from the interval $[0, 1]$. In general, modern computers use 64 bits to represent real numbers.

But do we really need all these bits? Using 64 bit makes sense if we are talking about values of physical quantities – we want to preserve the accuracy with which these values are known. However, for degrees provided by experts this does not make much sense. Probably an expert can meaningfully distinguish between degrees 0.6 and 0.8, but realistically, we cannot expect anyone to meaningfully distinguish, e.g., between degrees 0.80 and 0.81.

How many different degrees of confidence do we actually need? It is known that people can meaningfully divide objects into no more that 7 plus minus two categories (see, e.g., [4, 7]). This means that to adequately capture

human opinions, it is sufficient to use between $7 - 2 = 5$ and $7 + 2 = 9$ different degrees of confidence.

In general, absolutely true and absolutely false are also degree of confidence – so they are among these ≤ 9 degrees. However, experts are practically never absolutely 100% sure that the statement is true, and practically never absolutely sure that the statement is false. If we exclude these two degrees – absolutely true and absolutely false – we end up with no more than 7 different possible expert's degrees of certainty.

So how many bits per number do we need? To represent these degrees, we only need 3 bits, since using 3 bits allows us to represent $2^3 = 8$ different degrees of confidence.

Thus, to process each fuzzy degree on a mobile device, it is sufficient to use only 3 bits.

Comment. It it worth mentioning that the number of different distinguishable degrees is equal to the number of different basic colors. This is not a coincidence, since both numbers come from the same general seven plus minus two law. It is therefore possible to place different degrees of confidence in 1-to-1 correspondence with colors.

This is not just a purely mathematical possibility: such a correspondence enables us to effectively use optical computing – namely, a special color version of it – to speed up fuzzy computations; see, e.g., [8, 9, 10].

6 General Conlusions

To minimize energy consumption when performing fuzzy computations on a mobile device, we need:

- to use piecewise-linear membership functions (e.g., triangular and trapezoidal),
- to use min as "and"-operation, max as "or"-operation, and $1 - a$ as negation operation, and
- to use 3-bit representations of all fuzzy degrees.

Good news is that many effective applications of fuzzy techniques already use these membership functions and these logic operations, so their use should not lead to a drastic decrease in the quality of the results.

Acknowledgments

This work was supported in part by the National Science Foundation grants 1623190 (A Model of Change for Preparing a New Generation for Professional Practice in Computer Science), HRD-1834620 and HRD-2034030 (CAHSI Includes), EAR-2225395 (Center for Collective Impact in Earthquake Science C-CIES), and by the AT&T Fellowship in Information Technology.

It was also supported by a grant from the Hungarian National Research, Development and Innovation Office (NRDI).

References

[1] R. Belohlavek, J. W. Dauben, and G. J. Klir, *Fuzzy Logic and Mathematics: A Historical Perspective*, Oxford University Press, New York, 2017.

[2] G. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic*, Prentice Hall, Upper Saddle River, New Jersey, 1995.

[3] J. M. Mendel, *Uncertain Rule-Based Fuzzy Systems: Introduction and New Directions*, Springer, Cham, Switzerland, 2017.

[4] G. A. Miller, "The magical number seven plus or minus two: some limits on our capacity for processing information", *Psychological Review*, 1956, Vol. 63, No. 2, pp. 81–97.

[5] H. T. Nguyen, C. L. Walker, and E. A. Walker, *A First Course in Fuzzy Logic*, Chapman and Hall/CRC, Boca Raton, Florida, 2019.

[6] V. Novák, I. Perfilieva, and J. Močkoř, *Mathematical Principles of Fuzzy Logic*, Kluwer, Boston, Dordrecht, 1999.

[7] S. K. Reed, *Cognition: Theories and Application*, SAGE Publications, Thousand Oaks, California, 2022.

[8] V. Timchenko, Y. Kondratenko, and V. Kreinovich, "Efficient optical approach to fuzzy data processing based on colors and light filter", *International Journal of Problems of Control and Informatics*, 2022, Vol. 52, No. 4.

[9] V. Timchenko, Y. Kondratenko, and V. Kreinovich, "Decision support system for the safety of ship navigation based on optical color logic gates", *Proceedings of the IX International Conference "Information Technology and Implementation IT&I-2022*, Kyiv, Ukraine, November 30 — December 2, 2022.

[10] V. Timchenko, Y. Kondratenko, and V. Kreinovich, "Implementation of optical logic gates based on color filters", *Proceedings of the 6th International Conference on Computer Science, Engineering and Education Applications ICCSEEA 2023*, Warsaw, Poland, March 17–19, 2023.

[11] L. A. Zadeh, "Fuzzy sets", *Information and Control*, 1965, Vol. 8, pp. 338–353.