How to Explain (and Overcome) 2% Barrier in Teaching Computer Science: Fuzzy Ideas Can Help

Olga Kosheleva[1] and Vladik Kreinovich[2]
[1]Department of Teacher Education
[2]Department of Computer Science
University of Texas at El Paso
500 W. University
El Paso, TX 79968, USA
olgak@utep.edu. vladik@utep.edu

Abstract

Computer science educators observed that in the present way of teaching computing, only 2% of students can easily handle computational concepts – and, as a result, only 2% of the students specialize in computer science. With the increasing role of computers in the modern world, and the increasing need for computer-related jobs, this 2% barrier creates a shortage of computer scientists. We notice that the current way of teaching computer science is based on easiness of using two-valued logic, on easiness of dividing all situations, with respect to each property, into three classes: yes, no, and unknown. The fact that the number of people for whom such a division is natural is approximately 2% provides a natural explanation of the 2% barrier – and a natural idea of how to overcome this barrier: to tailor our teaching to students for whom division into more than three classes is much more natural. This means, in particular, emphasizing fuzzy logic, in which for each property, we divide the objects into several classes corresponding to different degrees with which the given property is satisfied.

1 Introduction

2% barrier: a brief description. Computer science educators have observed that only about two out of every 100 students enrolling in introductory programming courses really resonate with the subject and seem to be natural-born computer scientists; see, e.g., [3, 5, 6, 7]. This observation is in good accordance with the fact that in many universities, computer science students form about 2% of the total number of students, both on the undergraduate and on the graduate level.

Donald E. Knuth, one of the world leading computer scientists and computer science educators, uses this know fact to conclude that "roughly 2% of all people 'think algorithmically', in the sense that they can rapidly reason about algorithmic processes" [5, 6, 7].

Why the 2% barrier is a problem. As the world becomes more and more computerized, the society needs more and more computer scientists – or at least people who can think algorithmically. At present, there is a shortage of computer scientists – and in spite of the numerous efforts by computer science programs, the number of computer science students increases too slowly to cover this shortage.

What we do in this paper. How can we overcome this 2% barrier? A natural way to solve a problem is to analyze it. For the 2% problem, this means that we need to first understand the reasons behind the 2% barrier. In this paper, we provide a possible explanation for this barrier, and use this explanation to describe a possible oath to solving this problem.

2 Why 2% Barrier? A Possible Explanation

What computational thinking means now? In most computer science education programs, computational thinking includes the ability to take imprecise problems and reformulate them in precise terms, in terms which make it easier to explain these problems in a language that a computer can understand. A large part of this thinking is related to the traditional crisp 2-valued logic, whether it is the actual logic with "and", "or", and "not" used in programming languages, whether it is the fact that in the computer, everything has to be represented as a sequence of 0s and 1s.

What crisp computational thinking means in psychological terms. For people to use this logic, they need to be able, with respect to each property, to easily classify objects into three categories:

- objects which satisfy this property,
- objects which do not satisfy this property, and
- objects about which we do not know whether they satisfy the given property or not.

What is known about people's ability to easily classify into classes. Classification of objects into classes is a process well-studied by psychologists. A well-known "7 plus minus 2 law" states that people are most comfortable with classifying into 7 ± 2 classes; see, e.g., [8, 9]. This general psychological law has also been confirmed in our specific area of formalizing expert knowledge: namely, in [1, 2], it was shown that this law explains why in intelligent control,

experts normally use ≤ 9 different degrees (such as "small", "medium", etc.) to describe the value of each characteristic (see also [13]).

What the 7 ± 2 law tells us about the number of people who easily classify into 3 classes. What is the precise meaning of the 7 ± 2 law? This law does not mean that for *all* people, the number of classes into which they naturally classify is always between $7-2=5$ and $7+2=9$: there are people for whom the natural number of classes is smaller than 5, and there are people for whom the natural number of classes is larger than 9. A natural way to interpret this law is to treat it the same way as $\pm$ notations in science and engineering are usually interpreted (see, e.g., [11]): namely, to understand this law as saying that the among the human population, the mean value of the natural number of classes is 7 and the standard deviation is 2.

As in many other real-life sitiuations, the number of classes is affected by many different factors; in such situations, it is reasonable to apply the Central Limit Theorem, according to which the joint effect of many independent effects leads (under some reasonable conditions) to a normal distribution; see, e.g., [12]. Thus, it is reasonable to conclude that the natural number of classes is a normally distributed random variable with mean $\mu = 7$ and standard deviation $\sigma = 2$.

Based on these ideas, what is the proportion of people for whom computational thinking is natural? As we have mentioned, computational thinking means that it is natural for a person to divide everything into $X = 3$ classes – or even sometimes into 2 classes ("yes" and "no"). In other words, computational thinking means that the natural number of classes does not exceed 3: $X \le 3$.

Now that we know the probability distribution for the natural number of classes X, we can estimate the probability $P(X \le 3)$. A usual way of computing such probabilities for a normal distribution with given mean μ and standard deviation μ is to reduce it to the *standard* normal distribution $Z \stackrel{\text{def}}{=} \dfrac{X-\mu}{\sigma}$, with 0 mean and standard deviation 1. For each real number x, the inequality $X \le x$ is equivalent to $Z \le z \stackrel{\text{def}}{=} \dfrac{x-\mu}{\sigma}$. Thus, the desired probability $P(X \le x)$ is equal to $\Psi(z)$, where $\Psi(z) \stackrel{\text{def}}{=} P(Z \le z)$ is a (well-tabulated) cumulative distribution function for the standard normal distribution.

In our case, we have $x = 3$, $\mu = 7$, and $\sigma = 2$, so $z = \dfrac{x-\mu}{\sigma} = \dfrac{3-7}{2} = -2$ and thus, $P(X \le 3) = \Phi(-2) \approx 2.3\%$.

These computations explains the 2% barrier. The above computations show that approximately 2% of people have the ability to easily classify all the objects into three classes (yes, no, and unknown), an ability which is crucial to computational thinking as it is taught now.

3 How Can We Overcome the 2% Barrier? Fuzzy Ideas Can Help

Our explanation of the 2% barrier: reminder. Our explanation seems to indicate the origin of the 2% barrier to increasing the number of computer scientists:

- a traditional way to study computer science is based on emphasizing two-valued logic, while
- for 98% of the people, this logic is *not* very natural.

How to overcome this barrier? From this viewpoint, a reasonable way to overcome the 2% barrier is to do more to tailor computer science education to folks for whom division into more than 3 classes is much more natural. In particular, instead of focusing on the two-valued logic, such tailored approaches should emphasize multiple-valued logic.

Fuzzy ideas can help. In principle, from the mathematical viewpoint, there are many different multiple-valued logics. Which one should we choose?

A good criterion for selection is taking into account that applications are always a good stimulus for learning. It is reasonable to select a multiple-valued logic which has the largest number of practical applications – and this is undoubtedly *fuzzy logic*; see, e.g., [4, 10, 14].

Conclusion. So, we arrive at the following conclusion: to overcome the 2% barrier, to satisfy the society's need for computer scientists, to increase the number of computer scientists, we need to introduce fuzzy logic as early as possible into the computer science education.

Discussion. Of course, simply introducing fuzzy logic is not a panacea, we need to go further and reemphasize all the usual binary concepts of computer science in terms of scales consisting of 7 ± 2 elements. Some of this reformulation has already been done in fuzzy research: there are notions of (and results about) fuzzy automata, fuzzy graphs, even fuzzy algorithms. However, all this is just the beginning, the main work is still ahead.

Acknowledgment

This work was supported in part by the National Science Foundation grants HRD-0734825 and HRD-1242122 (Cyber-ShARE Center of Excellence) and DUE-0926721, by Grants 1 T36 GM078000-01 and 1R43TR000173-01 from the National Institutes of Health, and by a grant on F-transforms from the Office of Naval Research.

References

[1] P. Bonissone and K. Decker, "Selecting Uncertainty Calculi and Granularity: An Experiment in Trading-off Precision and Complexity", In: L. N. Kanal and J. F. Lemmer (eds.), *Uncertainty in Artificial Intelligence*, North Holland, Amsterdam, 1986, pp. 217–247.

[2] L. Godo, R. Lopez de Mantaras, C. Sierra, and A. Verdaguer, "MILORD: The Architecture and management of Linguistically expressed Uncertainty", *International Journal of Intelligent Systems*, 1989, Vol. 4, pp. 471–501.

[3] F. Gruenberger, "The role of education in preparing effective computing personnel," in: F. Gruenberger (ed.), *Effective vs. Efficient Computing*, Prentice-Hall, Englewood Cliffs, New Jersey, 1973, pp. 112–120.

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

[5] D. E. Knuth, "Algorithms in modern mathematics and computer science", *Proceedings of the International Symposium on Algorithms in Modern mathematics and Computer Science, Khiva, Uzbekistan, September 1979*, Springer Lecture Notes in Computer Scinece, 1981, Vol. 122, pp. 82–99.

[6] D. E. Knuth, "Algorithmic thinking and mathematical thinking", *American Mathematical Monthly*, 1985, Vol. 92, pp. 170–181.

[7] D. E. Knuth, *Selected Papers on Computer Science*, Camridge University Press, 1996.

[8] G. A. Miller, "The magical number seven plus or minus two: some limits on our capacity for processing information", *Psychological Review*, 1956, Vol. 63, pp. 81–97.

[9] P. M. Milner, *Physiological psychology,* Holt, NY, 1970.

[10] H. T. Nguyen and E. A. Walker, *First Course In Fuzzy Logic*, CRC Press, Boca Raton, Florida, 2006.

[11] S. Rabinovich, *Measurement Errors and Uncertainties: Theory and Practice*, American Institute of Physics, New York, 2005.

[12] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, Chapman and Hall/CRC Press, Boca Raton, Florida, 2011.

[13] R. Trejo, V. Kreinovich, I. R. Goodman, J. Martinez, and R. Gonzalez, "A Realistic (Non-Associative) Logic And a Possible Explanations of 7 ± 2 Law", *International Journal of Approximate Reasoning*, 2002, Vol. 29, pp. 235–266.

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