“At Least k out of n” under Fuzzy Uncertainty: Efficient Algorithm for General “And”-Operations

Olga Kosheleva[1], Vladik Kreinovich[2], and Klaus-Peter Adlassnig[3]

[1] Department of Teacher Education, University of Texas at El Paso,
500 W. University, El Paso, Texas 79968, USA, olgak@utep.edu,
https://www.cs.utep.edu/vladik/olgavita.html

[2] Department of Computer Science, University of Texas at El Paso,
500 W. University, El Paso, Texas 79968, USA, vladik@utep.edu,
https://www.cs.utep.edu/vladik/

[3] Medical University of Vienna, Center for Medical Data Science,
Institute of Artificial Intelligence, Spitalgasse 23, 1090 Vienna, Austria, and
Medexter Healthcare, Borschkegasse 7/5, 1090 Vienna, Austria
klaus-peter.adlassnig@meduniwien.ac.at

Abstract. In medicine, many diagnoses are made when, for some value k, at least k of n possible symptoms are present. Many of such symptoms – such as fever – are, in reality, fuzzy. For example, it makes no sense that say that 38.0 is fever while 37.9 is not a fever, both are fever to some degree. Once such degrees are given, we need to use them to estimate the degree to which the patient has the corresponding disease. For this problem, the usual fuzzy techniques require exponentially many computational steps – so it is desirable to have a more efficient algorithm. Such an algorithm was previously proposed for some specific “and”-operation (t-norm). However, in different application areas, different “and”-operation describe the reasoning within this domain. So, it is desirable to extend the existing feasible algorithm to the case of general “and”-operations. In this paper, we describe such an extension.

Keywords: fuzzy logic, at least k out of n, medical diagnostics

1 Formulation of the problem

“At least k out of n” is practically useful. In medicine, often a diagnosis is based on the condition that a certain number of symptoms are present – at least k symptoms out of n possible symptoms, for some k. Let us give two examples from [2]:

– “A patient is classified as high risk for septic shock if they show at least four out of the following signs: hypotension, tachycardia, fever, leukocytosis, altered mental status, elevated serum lactate levels.”

– "A patient is classified as low risk for septic shock if they show at most two out of the following signs: hypotension, tachycardia, fever, leukocytosis, altered mental status, elevated serum lactate levels." – which means that when there are at least three signs, the risk is medium or high.

Need to take fuzzy-type uncertainty into account. In principle, one can use "yes"-"no" (crisp) definitions of the symptoms. For example, we can say that a body temperature of 38 C or higher is fever, while anything smaller than 38 is not a fever.

However, from the commonsense viewpoint, it does not make sense to call 38.0 a fever and 37.9 not a fever: the difference between the two values is close to the accuracy of the thermometer. From this viewpoint, it makes more sense to talk about to what degree the patient has a fever:

– temperature much smaller than 38 means that this is definitely not a fever,
– temperature much larger than 38 means that this is definitely a fever, but
– temperature close to 38 mean that we have a fever to some degree.

The technique to deal with such degrees is what is usually called *fuzzy logic*; see, e.g., [1, 4, 6, 8–10]. In these terms, what we need is to take into account fuzzy uncertainty.

How to estimate to what extent is the at-least-k-out-of-n condition satisfied: a natural idea. When each statement is either true or false, it is straightforward to decide when we have at least k out of n symptoms. However, when we have symptoms satisfied to certain degrees $m_1, \ldots, m_n$, the at-least-k-out-of-n condition is only satisfied to some degree. How can we compute this degree?

In fuzzy techniques, a usual way to assign a degree to a complex statement is:

– to represent this statement in terms of the basic logical connectives – "and", "or", and "not" – and then
– to use fuzzy analogs of these connectives – i.e., "and"-operations $f_\&(a,b)$ (also known as t-norms), "or"-operations $fvee(a,b)$ (also known as t-conorms), and negation operations $f_\neg(a)$.

For example, "at least k out of n" means that the set of symptoms can be any set S with at least k elements.

– For each of these sets, the above approach leads to the formula

$$m(S) = f_\&(m_{i_1}, \ldots, n_{i_\ell}, f_\neg(m_{j_1}), \ldots, f_\neg(m_{j_p})),$$

where $i_1, \ldots, i_\ell$ are all elements of the set S, while $j_1, \ldots, j_p$ are all symptoms that do not belong to the set S.
– Then, the degree m to which the condition is satisfied can be computed as $f_\vee(m(S_1), m(S_2), \ldots)$, where S_1, S_2, etc. are all the subsets of the set $\{1, 2, \ldots, n\}$ that have at least k elements.

Limitations of the above natural idea. In the above natural-idea approach, to find the desired degree m, we need to compute $m(S)$ for a large number of sets – close to 2^n. For large n, this number becomes astronomical.

Need to be efficient. Because of this limitation, it is desirable to come up with an efficient way to define and compute the desired degree.

What is known. For the case when the "and"-operation is

$$f_\&(a, b) = \max(a + b - 1, 0),$$

an efficient algorithm for estiating the desired degree m was proposed in [5].

Remaining problem: need to consider general "and"-operations. We want our degree to reflect the practice of the corresponding discipline – e.g., of the corresponding branch of medicine. In general, reasoning in different branch of knowledge is best described by different "and"-operations.

This was first discovered when the first expert systems appeared. Historically the first expert system was the system MYCIN that was focused on a certain class of blood diseases; see, e.g., [3]. Designers of this system spent a lot of time and efforts searching for the "and"-operation that provides the most accurate description of the reasoning of the medical doctors dealing with these diseases. At first, they were under the impression that they found universal laws of human reasoning. However, when they tried to apply the same "and"-operation to a different application area – geophysics – they quickly found out that the resulting degrees for "and"-statements were very different from what the actual geophysicists produced – and thus, that a different "and"-operation is needed for geophysical applications.

This difference in "and"-operations make perfect sense:

- in medicine, one needs to be very cautious, and to prescribe some cure only if we are reasonable sure that it will help – mistakes can deadly;
- in contrast, in mining applications of geophysics, if a company waits too long for a perfect conformation that there is oil in a field, it may lose to competitors – too much caution can ruin a company.

Since we want to take care of all possible applications where the "at least k out of n" idea is used, we need to extend the above-mentioned feasible algorithm for a *specific* "and"-operation to the case of *general* "and"-operations.

What we do in this paper. In this paper, we solve this problem by providing an efficient algorithm for general "and"-operations.

2 Our solution

How can we describe a general "and"-operation: a brief reminder. It is known (see, e.g., [7]) that any continuous "and"-operation can be approximated, with any given accuracy $\varepsilon > 0$, by a *strictly Archimedean* "and"-operation, i.e., operation of the type

$$f_\&(a,b) = f^{-1}(f(a) \cdot f(b)) \quad (1)$$

for some continuous strictly increasing function $f(a)$. Here, $f^{-1}(a)$ means the inverse function, i.e., the function for which $f^{-1}(b) = a$ if and only if $f(a) = b$.

Since, based on finitely many experiments with real experts, we can only determine the "and"-operation with some accuracy, this means that we can have a strictly Archimedean "and"-operation that is in perfect accordance with the experimental data. Thus, to describe actual expert reasoning, we can always use strictly Archimedean "and"-operations.

Our first idea. The use of strictly Archimedean "and"-operation means, in effect, that:

– if we re-scale degrees, i.e., replace each degree m_i with the degree $m'_i = f(m_i)$, then we can simply use multiplication as "and", and
– at the end of computations, we need to transform the resulting degree m' back to the original scale by taking $m = f^{-1}(m')$.

So this is our idea of how to estimate the desired degree m under a general strictly Archimedean "and"-operation (1):

– first, we transform the original degrees m_i into re-scaled ones $m'_i = f(m_i)$,
– then, we use the values m'_i to perform multiplication-based estimation of the degree m, and
– after that, compute the desired estimate m as $f^{-1}(m)$.

Our second idea: how to perform multiplication-based computations. We have reduced the problem for a general "and"-operation to its specific case, when the "and"-operation is simply multiplication. So, to solve the general problem, it is sufficient to solve it for the case of the product "and"-operation $f_\&(a,b) = a \cdot b$.

For this operation, the "and"-formula is similar to the probabilistic case, where for two independent events, the probability that both will occur is equal to the product $a \cdot b$ of the probabilities a and b of each of the two events. Probability theory exists for many centuries, many algorithms have been designed for it. So, for the multiplication "and"-operation, a natural way to provide an estimate is to view the values m'_i as probabilities and to estimate the probability that at least k out of n symptoms are satisfied.

To use this idea, we need to come up with the efficient algorithm for computing this probability.

Comment. Of course, it is important to emphasize that the use of probabilities is simply a mathematical trick, it does not mean that expert-produced degrees m_i somehow became probabilities.

How to effectively compute the desired probability: analysis of the problem. The probability m' that at least k out of n events occur can be described as

$$m' = 1 - p_1 - p_2 - \ldots - p_{k-1}, \tag{2}$$

where p_j is the probability that exactly j events happened. So, to compute m', it is sufficient to be able to estimate the values p_j.

One can easily check that we have

$$p_j = \sum_{S:|S|=j} \left(\prod_{i\in S} m'_i \cdot \prod_{i\notin S} (1 - m'_i) \right), \tag{3}$$

where $|S|$ denotes the number of elements in the set S. Each term in this sum can be equivalently described as

$$\prod_{i\in S} m'_i \cdot \prod_{i\notin S} (1 - m'_i) = p \cdot \prod_{i\in S} r_i, \tag{4}$$

where

$$p \stackrel{\text{def}}{=} \prod_{i=1}^{n} m'_i \tag{5}$$

and

$$r_i \stackrel{\text{def}}{=} \frac{m'_i}{1 - m'_i}. \tag{6}$$

Thus, we have

$$p_j = \sum_{S:|S|=j} \left(p \cdot \prod_{i\in S} r_i \right). \tag{7}$$

All the terms in the sum have a common factor, thus we can take this factor out of the sum and get:

$$p_j = p \cdot s_j, \text{ where } s_j \stackrel{\text{def}}{=} \sum_{S:|S|=j} \prod_{i\in S} r_i. \tag{8}$$

So, the question is how to compute the values s_j.

For $j = 1$, we simply have

$$s_1 = \sum_{i=1}^{n} r_i. \tag{9}$$

Once we know s_j, we can try to help to compute s_{j+1} by multiplying s_j and s_1. Both s_j and s_1 are sums corresponding to sets S of k elements and a set of a single element.

- When the single element is not in S, we get the product corresponding to their union – set of $k + 1$ elements.
- But when the single element *is* in S, then we get a product in which the ratio r_i corresponding to this element is repeated twice, i.e., we have this ratio squared.

In other words, when we multiply s_j and s_1, we get the sum s_j and some auxiliary term. When we try to get this term for j in terms of a similar term for $j-1$, we will get an auxiliary term in which in each product, one of the probabilities is cubed, etc.

In general, we therefore need to compute the terms

$$s_{j,a} \stackrel{\text{def}}{=} \sum_{S:|S|=j} \sum_{i \in S} \left(r_i^a \cdot \prod_{\ell \neq i, \ell \in S} r_\ell \right). \tag{10}$$

For $j = 1$, we simply have

$$s_{1,a} = \sum_{i=1}^{n} r_i^a. \tag{11}$$

In these terms, for s_j, we have

$$s_j = s_{j-1} \cdot s_1 - s_{j-1,2}. \tag{12}$$

In general, similarly to what we discussed a little earlier, we get the following equality:

$$s_{j,a} = s_{j-1} \cdot s_{1,a} - s_{j-1,a+1}. \tag{13}$$

So, according to (12), to compute s_j, we need to compute $s_{j-1,2}$. To compute this terms, according to (13), we need to compute $s_{j-2,3}$, etc., until we get $s_{1,a'}$ for some a' – which is easy to compute.

So, we arrive at the following algorithm.

3 Resulting algorithm

What is given. We know the degrees $m_1, \ldots, m_n$ of each of n symptoms. We also know the function $f(x)$ for which the "and"-operation $f^{-1}(f(a), f(b))$ best describes the reasoning of people from this particular application area.

What we want. We want to estimate the degree to which, based on this information, at least k symptoms are present.

Preliminary stage. We compute the values $m_i' = f(m_i)$, $r_i = m_i'/(1 - m_i')$, and $p = (1 - m_1') \cdot (1 - m_n')$.

Main stage.

- We compute the values $s_{1,a} = (r_1)^a + \ldots + (r_n)^a$ for $a = 1, \ldots, k-1$. In particular, for $a = 1$, we get $s_1 = s_{1,1}$.
- Then, for $j = 2, \ldots, k$, we do the following:
  - we use the formula (13) to sequentially compute the values $s_{2,j-1}$, $s_{3,j-2}$, $\ldots$, $s_{j-1,2}$ and then
  - we use formula (12) to compute s_j.

Final stage. We compute $m' = 1 - p \cdot (s_1 + \ldots + s_{k-1})$ and $m = f^{-1}(m')$.

Comment. One can check that:

- on the preliminary stage, we perform $O(n)$ steps;
- on the main stage, we perform $O(k \cdot n)$ steps; and
- on the final stage, we perform $O(k)$ steps.

So, the overall number of steps is $O(k \cdot n)$. In particular, for any given k, this algorithm requires computation time that is linear in n.

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), and by the Institute for Risk and Reliability, Leibniz Universitaet Hannover, Germany.

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. J. S. de Bruin, K. P. Adlassnig, A. Blacky, and W. Koller, "Detecting borderline infection in an automated monitoring system for healthcare-associated infection using fuzzy logic", *Artificial Intelligence in Medicine*, 2016, Vol. 69, pp. 33–41, doi: 10.1016/j.artmed.2016.04.005.
3. B. G. Buchanan and E. H. Shortliffe, *Rule Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*, Addison-Wesley, Reading, Massachusetts, 1984.
4. G. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic*, Prentice Hall, Upper Saddle River, New Jersey, 1995.
5. O. Kosheleva, V. Kreinovich, and H. P. Nguyen, "How to Describe Conditions Like 2-out-of-5 in Fuzzy Logic: A Neural Approach", *Journal of Advanced Computational Intelligence and Intelligent Informatics JACIII*, 2020, Vol. 24, No. 5, pp. 593–598, doi: 10.20965/jaciii.2020.p0593
6. J. M. Mendel, *Explainable Uncertain Rule-Based Fuzzy Systems*, Springer, Cham, Switzerland, 2024.
7. H. T. Nguyen, V. Kreinovich, and P. Wojciechowski, "Strict Archimedean t-norms and t-conorms as universal approximators", *International Journal of Approximate Reasoning*, 1998, Vol. 18, Nos. 3–4, pp. 239–249.
8. H. T. Nguyen, C. L. Walker, and E. A. Walker, *A First Course in Fuzzy Logic*, Chapman and Hall/CRC, Boca Raton, Florida, 2019.
9. V. Novák, I. Perfilieva, and J. Močkoř, *Mathematical Principles of Fuzzy Logic*, Kluwer, Boston, Dordrecht, 1999.
10. L. A. Zadeh, "Fuzzy sets", *Information and Control*, 1965, Vol. 8, pp. 338–353.