# In Its Usual Formulation, Fuzzy Computation Is, In General, NP-Hard, But a More Realistic Formulation Can Make It Feasible

1st Martine Ceberio
*Department of Computer Science*
*University of Texas at El Paso*
El Paso, Texas 79968, USA
mceberio@utep.edu

2nd Olga Kosheleva
*Department of Teacher Education*
*University of Texas at El Paso*
El Paso, Texas 79968, USA
olgak@utep.edu

3rd Vladik Kreinovich
*Department of Computer Science*
*University of Texas at El Paso*
El Paso, Texas 79968, USA
vladik@utep.edu

4th Luc Longpré
*Department of Computer Science*
*University of Texas at El Paso*
El Paso, Texas 79968, USA
longpre@utep.edu

***Abstract*—In many practical situations, we cannot directly measure or estimate the desired quantity $y$ – e.g., we cannot directly measure the distance to a star or the next week's temperature. To provide the desired estimate, we measure or estimate easier-to-measure quantities $x_1, \ldots, x_n$ which are related to $y$, and then use the known relation to transform our estimates for $x_i$ into an estimate for $y$. In situations when $x_i$ are known with fuzzy uncertainty, we thus need *fuzzy computation*. Zadeh's extension principle provides us with formulas for fuzzy computation. The challenge is that the resulting computational problem is NP-hard – which means that, unless P=NP (which most computer scientists consider to be impossible), it is not possible to solve all fuzzy computation problems in feasible time. To overcome this challenge, we propose a more realistic formalization of fuzzy computation – in which instead of an un-realistic requirement that the corresponding properties hold for all $x_i$, we only require that they hold for almost all $x_i$ – in some reasonable sense. We show that under this modification, the problem of fuzzy computation becomes computationally feasible.**

***Index Terms*—fuzzy computation, Zadeh extension principle, interval computation, feasible, NP-hard, Monte-Carlo techniques**

## I. Introduction

For situations when a quantity $y$ depends, in a known way, on quantities $x_1, \ldots, x_n$, Zadeh's extension principle leads to useful formulas for computing the membership function for $y$ based on membership functions for $x_i$. However, the challenge is that the corresponding computational problem is NP-hard. This paper presents a realistic modification of Zadeh's extension principle, in which properties related to input variables can be only true to some extent. The paper shows that this modification makes it possible to design a feasible algorithm for solving the corresponding fuzzy computation problem.

This work was supported in part by the National Science Foundation grant HRD-1242122 (Cyber-ShARE Center of Excellence).

## II. Formulation of the Problem

**Need for computations.** To explain our problem, let us recall why we need computations in the first place. The main objectives of science and engineering are:

- to describe the world,
- to predict what will happen in the future, and,
- if necessary, to come up with recommendation of what to do to make the future state of the world better.

The physical world is usually described by the values of the corresponding physical quantities. Thus, to describe the current state of the world, we need to describe the numerical values of all these quantities.

Some of these values we can direct measure or estimate. For example:

- we can directly measure the width of a room;
- by touching a baby's forehead, we can directly estimate the baby's body temperature, etc.

However, there are many other quantities which are difficult to measure or estimate directly. For example, it is not easy to directly measure or estimate:

- the distance to a faraway star, or
- the temperature inside the car engine.

This impossibility is even more evident if we are interested in the future values of the quantities of interest: there is no way that we can measure them now, and it is rarely possible to directly estimate these future values – otherwise, we would not need high-performance computers to predict next week's weather.

If we cannot directly measure or estimate the value of the desired quantity $y$, a natural idea is to estimate this value indirectly. Namely:

- we find some easier-to-or-estimate auxiliary quantities $x_1, \ldots, x_n$ which are related to the desired quantity $y$ by a known dependence $y = f(x)$, where we denoted
$$x \stackrel{\text{def}}{=} (x_1, \ldots, x_n); \qquad (1)$$
- then, we measure or estimate these auxiliary quantities;
- finally, we use the resulting estimates $\widetilde{x}_i$ to compute the estimate $\widetilde{y} = f(\widetilde{x}_1, \ldots, \widetilde{x}_n)$ for the desired quantity $y$.

For example, to predict the temperature $y$ in El Paso in a week, we can:

- measure the values $x_1, \ldots, x_n$ describing the temperature, humidity, and wind speed measurements in a wide area, and then
- use the algorithm $y = f(x_1, \ldots, x_n)$ for solving the corresponding partial differential equations to predict how the next-week temperature in El Paso.

Such estimations are the main reason why computations are needed in the first place.

**Need for fuzzy computation.** This subsection is intended for readers from outside fuzzy community – e.g., for specialists in computational complexity who may be interested in our result. Readers who know the basics of fuzzy can skip this subsection.

In many situations, the values $x_i$ are estimated by experts – or measured by measuring instruments whose uncertainty is described by experts. Experts usually describe the accuracy of their estimates not in precise mathematical terms, but rather by using imprecise ("fuzzy") words from natural language. For example, an expert can say that the value of $x_i$ is "close to 1", "the difference between the actual value and 1 is small", or, more accurately, "close to 1 with an estimation error around 0.1". To use these imprecise estimates, we need to translate them into precise computer-understandable terms.

For this translation, it is reasonable to use fuzzy logic – a technique specifically designed by Lotfi Zadeh for the purpose of such a translation; see, e.g., [2], [5], [8], [10], [12], [24]. In fuzzy logic, each imprecise property (like "small") is described by assigning, to each possible value of the corresponding quantity $q$, the degree $\mu(q) \in [0,1]$ to which this value satisfies the corresponding property – e.g., is small. The corresponding function $q \to \mu(q)$ is known as the *membership function*.

After this translation, we have $n$ membership functions $\mu_1(x_1), \ldots, \mu_n(x_n)$. Based on these functions, we would like to come up with the membership function $\mu(y)$ describing possible values of $y = f(x_1, \ldots, x_n)$. Computing this function is a fuzzy case of the general computation problem; we will therefore call computing $\mu(y)$ *fuzzy computing*.

**Traditional formulas for fuzzy computing: Zadeh's extension principle.** How can we compute $\mu(y)$? The main idea behind this computation was proposed by Zadeh himself. This idea is based on a fact that the number $Y$ is a possible value of the quantity $y = f(x_1 \ldots, x_n)$ if $Y = f(X_1, \ldots, X_n)$ for some possible values $X_i$ of $x_i$. In other words, $Y$ is possible if:

- either $X_1$ is a possible value of $x_1$ *and* $X_2$ is a possible value of $x_2$, *and* ..., for some tuple $(X_1, \ldots, X_n)$ for which $Y = f(X_1, \ldots, X_n)$,
- *or* $X'_1$ is a possible value of $x_1$ *and* $X'_2$ is a possible value of $x_2$, *and* ..., for some tuple $(X'_1, \ldots, X'_n)$ for which $Y = f(X'_1, \ldots, X'_n)$,
- or the same us true for other values $X''_1, \ldots, X''_n$.

For each $i$ and for each value $X_i$, we know the degree to which $X_i$ is a possible value of $x_i$ – this value is $\mu_i(X_i)$, by definition of the membership function. So, to find the expert's degree of confidence in the above composite statement, we need to be able, in general, to combine the degrees of confidence in two statements $S$ and $S'$ into a degree of confidence in a propositional combination "$S$ and $S'$" or "$S$ or $S'$".

In fuzzy logic, such a combination is provided by, correspondingly, "and"-operations (also known as *t-norms*) and "or"-operations (also known as *t-conorms*). The simplest such operations are $\min(a, b)$ and $\max(a, b)$. By using these operations, we get the following formula for the desired degree $\mu(Y)$:
$$\mu(Y) = \max_{X_1, \ldots, X_n : Y = f(X_1, \ldots, X_n)} \min\{\mu_1(X_1), \mu_2(X_2), \ldots\}. \qquad (2)$$
This formula, first proposed by Zadeh, is known as *Zadeh's extension principle*, since it extends the function $f(x_1, \ldots, x_n)$ used in traditional numerical computations to the case when inputs are fuzzy; see, e.g., [2], [5], [8], [10], [12], [24].

**Challenge: in its usual formulation, fuzzy computation is, in general, NP-hard.** We have shown that fuzzy computation is practically important. How difficult is it to perform the corresponding computations?

As we will show, in general, fuzzy computations are NP-hard; see, e.g., [7], [22]. Crudely speaking, this implies that, if (as most computer scientists believe) P is different from NP, no algorithm is possible that would solve all fuzzy computation problems in feasible time [7], [14]. We will show that this is even true if we restrict ourselves to quadratic functions
$$f(x_1, \ldots, x_n) = a_0 + \sum_{i=1}^n a_i \cdot x_i + \sum_{i=1}^n \sum_{j=1}^n a_{ij} \cdot x_i \cdot x_j. \qquad (3)$$
For some inputs, the computation time grows exponentially with the number of inputs $n$, and thus, already for values $n \approx 50$, becomes larger than the lifetime of the Universe – i.e., completely unrealistic and infeasible.

In practical problems, the number of inputs can be huge. For example, when we estimate the quality of a road, we use the opinion of multiple pavement experts who provide estimates of different quality aspects of different road segments; see, e.g., [16], [18], [19].

The proof of NP-hardness is very straightforward. Namely, it is well known – and relatively easy to prove – that for the membership function $\mu(y)$, the corresponding $\alpha$-cut
$$\mathbf{y}(\alpha) \stackrel{\text{def}}{=} \{y : \mu(y) \geq \alpha\} \qquad (4)$$

is equal to the range of $f(x_1, \ldots, x_n)$ when each $x_i$ is in the corresponding $\alpha$-cut $\mathbf{x}_i(\alpha) = \{x_i : \mu_i(x_i) \geq \alpha\}$:

$$\mathbf{y}(\alpha) = \{f(x_1, \ldots, x_n) : x_i \in \mathbf{x}_i(\alpha) \text{ for all } i\}. \qquad (5)$$

It is known that even in the simplest case, when the sets $\mathbf{x}_i(\alpha)$ are intervals, computing the range is NP-hard for quadratic functions $f(x_1, \ldots, x_n)$ [7], [22].

*Comment.* This results means that, unless P = NP, no general feasible algorithm is possible for performing fuzzy computations: for any fuzzy computations algorithm, time complexity grows very fast with the number of variables $n$.

It is worth mentioning that if we fix $n$ and increase the values $x_i$, then, e.g., for quadratic functions (or, in general, for polynomial dependence), the time complexity grows feasibly (i.e., polynomially); see, e.g., [7]. So, the problem is not the *size* of each input $x_u$; the problem is the *number* of inputs.

**Let us describe this result in precise terms.** For simplicity, let us consider the usual case when all $\alpha$-cuts are intervals, i.e., if $\mathbf{x}_i(\alpha) = [\underline{x}_i(\alpha), \overline{x}_i(\alpha)]$. The bounds $\underline{x}_i(\alpha)$ and $\overline{x}_i(\alpha)$ of these intervals must be given, i.e., must be represented in a computer – thus, we can safely assume that these bounds are rational numbers.

The function $f(x_1, \ldots, x_n)$ is continuous – as we have mentioned earlier, it is sufficient to consider quadratic functions. Similarly to the bounds of the $x$-intervals, we can safely assume that the coefficients of this quadratic functions are rational numbers.

Our goal is then to compute the range $\mathbf{y}(\alpha) = \left[\underline{y}(\alpha), \overline{y}(\alpha)\right]$. In general, the endpoint of a range are not necessarily rational numbers, so we may not be able to represent these ranges exactly. Thus, what we want, in general, is to generate rational numbers which are $\varepsilon$-close to the desired endpoints $\underline{y}(\alpha)$ and $\overline{y}(\alpha)$, for some given rational number $\varepsilon$.

By definition of the range, we have

$$\underline{y}(\alpha) = \min_{x_1 \in \mathbf{x}_1(\alpha), \ldots, x_n \in \mathbf{x}_n(\alpha)} f(x_1, \ldots, x_n) \qquad (6)$$

and

$$\overline{y}(\alpha) = \max_{x_1 \in \mathbf{x}_1(\alpha), \ldots, x_n \in \mathbf{x}_n(\alpha)} f(x_1, \ldots, x_n). \qquad (7)$$

We are considering the same value $\alpha$ in all these formulas. Thus, we can safely skip the mentioning of this value – this will make all the notations clearer without introducing any confusion. Thus, we arrive at the following formulation of the problem.

**Definition 1.** *By the* fuzzy computation problem*, we mean the following problem:*

*GIVEN: rational numbers $\underline{x}_1, \overline{x}_1, \ldots, \underline{x}_n, \overline{x}_n$, $\varepsilon > 0$, and a quadratic function $f(x_1, \ldots, x_n)$ with rational coefficients.*

*COMPUTE: rational numbers $\underline{r}$ and $\overline{r}$ which are $\varepsilon$-close to, correspondingly,*

$$\underline{y} = \min_{\underline{x}_1 \leq x_1 \leq \overline{x}_1 \,\&\, \ldots \,\&\, \underline{x}_n \leq x_n \leq \overline{x}_n} f(x_1, \ldots, x_n) \qquad (8)$$

*and*

$$\overline{y} = \max_{\underline{x}_1 \leq x_1 \leq \overline{x}_1 \,\&\, \ldots \,\&\, \underline{x}_n \leq x_n \leq \overline{x}_n} f(x_1, \ldots, x_n). \qquad (9)$$

**What we do in this paper.** As we have mentioned earlier, in general, he fuzzy computation problem is NP-hard. In this paper, we show that by having a more realistic formalization of fuzzy computing, we can make fuzzy computing feasible.

## III. Related Work

The relation between fuzziness and NP-hardness is well known and well exploited. Specifically, many authors have used fuzzy techniques to provide efficient algorithms for solving many particular cases of problems which are, in general, NP-hard; see, e.g., [1], [3], [4], [6], [13], [15], [17], [20], [23] and references therein.

This paper is different: instead of using fuzzy techniques to solve NP-hard problems, it shows how to modify a fuzzy computation problem so that it stops being NP-hard.

## IV. How to Make Formulas of Fuzzy Computing More Realistic

**Main idea.** The usual derivation of Zadeh's extension principle for quantum computing considers *all* possible tuples $(X_1, \ldots, X_n)$ for which $f(X_1, \ldots, X_n) = Y$. Similarly, in the formulas for the $\alpha$-cut, we consider *all* possible tuples $(x_1, \ldots, x_n)$ for which $\mu_i(x_i) \geq \alpha$ for every $i$. In both cases, we took "all" literally: all means all, one exception makes a statement about all the tuples false.

From the mathematical viewpoint, this is a reasonable idea. But let us take into account that we are not proving mathematical theorems, we are trying to formalize common sense, we are trying to formalize expert reasoning. And in our usual reasoning, "all" does not literally mean mathematically all, it usually means "almost all", meaning everyone except a small fraction of the original population.

- When a patriotic journalist says that all the citizens of the country support it against its enemy – real or perceived – this all-statement is often followed by a confession that there are a few traitors who are willing to support the enemy.
- When we say that all pigeons can fly, we understand very well that there may be a wounded or deformed pigeon, but that most pigeons can fly.
- A classical AI example is a phrase "all birds fly" (see, e.g., [11]); this phrase has known exceptions, such as penguins, but the vast majority of the birds indeed can fly.

Let us see how the above definitions of fuzzy computing will change if we replace the mathematical meaning of "all" with a more commonsense meaning of this word.

**Towards a new formalization of fuzzy computing.** In line with what we have just discussed, instead of defining the desired upper endpoint $\overline{y}$ as the exact maximum of all the values $f(x_1, \ldots, x_n)$ when $x_i \in [\underline{x}_i, \overline{x}_i]$, it is reasonable to define $\overline{y}$ as the maximum of "almost all" values.

If we fix the exact proportion $\delta > 0$ of values that we can ignore, that would mean that we are looking for a value $\overline{y}$ for which

$$\frac{|\{x : x_1 \in \mathbf{x}_1 \,\&\, \ldots \,\&\, x_n \in \mathbf{x}_n \,\&\, f(x_1, \ldots, x_n) \le \overline{y}\}|}{|\{x : x_1 \in \mathbf{x}_1 \,\&\, \ldots \,\&\, x_n \in \mathbf{x}_n\}|} = 1 - \delta, \tag{10}$$

where $|S|$ denotes the multi-D volume of a set $S$:

- width of an interval,
- area of a planar (2-D) set,
- volume of a 3-D set, etc.

When $\delta$ tends to 0, the corresponding value tends to the maximum of the function $f(x_1, \ldots, x_n)$ on the box

$$\mathbf{x} \stackrel{\text{def}}{=} \mathbf{x}_1 \times \ldots \times \mathbf{x}_n. \tag{11}$$

Thus, for small $\delta$, the above-defined value is very close to this maximum.

Similarly, instead of defining the lower endpoint $\underline{y}$ as the exact minimum of all the values $f(x)$ when $x \in \mathbf{x}$, it is reasonable to define $\underline{y}$ as the value for which

$$\frac{\left|\left\{x : x \in \mathbf{x} \,\&\, f(x) \ge \underline{y}\right\}\right|}{|\{x : x \in \mathbf{x}\}|} = 1 - \delta. \tag{12}$$

When $\delta$ tends to 0, the corresponding value tends to the minimum of the function $f(x_1, \ldots, x_n)$ on the box $\mathbf{x}$. Thus, for small $\delta$, the above-defined value is very close to this minimum.

What value $\delta$ should we choose? Intuitively, since we are considering the *fuzzy* case, it makes no sense to fix one *exact* value $\delta$, it is more appropriate to assume that this value is also given with some uncertainty – at least with interval uncertainty. In other words, it makes sense to assume that we know the interval $\left[\underline{\delta}, \overline{\delta}\,\right]$, with $\underline{\delta} < \overline{\delta}$, that contains the actual (unknown) value $\delta$. In this case, all we know about $1 - \delta$ is that it is somewhere between $1 - \overline{\delta}$ and $1 - \underline{\delta}$:

$$1 - \overline{\delta} \le 1 - \delta \le 1 - \underline{\delta}. \tag{13}$$

Thus, e.g., from the above equality for $\overline{y}$ we get the double inequality:

$$1 - \overline{\delta} \le \frac{|\{x : x \in \mathbf{x} \,\&\, f(x) \le \overline{y}\}|}{|\{x : x \in \mathbf{x}\}|} \le 1 - \underline{\delta}. \tag{14}$$

A similar inequality holds for $\underline{y}$.

**What does it mean to compute $\underline{y}$ and $\overline{y}$?** Since we relaxed the requirement on the endpoints $\underline{y}$ and $\overline{y}$ – by allowing some probability that these values are not mathematically universal bounds – it makes sense to also relax the usual requirement on the algorithm: that it always computes the desired value. Specifically, from the practical viewpoint, it makes perfect sense to consider algorithms that provide an answer with a probability $1 - p_0$, for some small positive value $p_0 \ll 1$.

Indeed, even the computer hardware is not 100% reliable, once in a while computers break down. From this viewpoint, it is perfectly OK if the algorithm also sometimes does not produce the desired result – as long as the probability for this is much smaller than the probability of a hardware fault and thus, does not significantly increase the probability of am erroneous result.

Thus, we arrive at the following definitions.

## V. Definitions of the More Realistic Fuzzy Computations and the Main Result: That the Corresponding Problem Is Now Feasible

**Definition 2.** *Let $\varepsilon > 0$ be a rational number. We say that a function $f(x_1, \ldots, x_n)$ is* $\varepsilon$-feasible *if there exists a feasible algorithm that, given rational values $x_1, \ldots, x_n$, produces a rational number which is $\varepsilon$-close to $f(x_1, \ldots, x_n)$.*

**Definition 3.** *Let $\varepsilon > 0$, $0 < \underline{\delta} < \overline{\delta}$, and $p_0 > 0$ be rational numbers. By* realistic fuzzy computations*, we mean the following problem:*

*GIVEN: rational numbers $\underline{x}_1, \overline{x}_1, \ldots, \underline{x}_n, \overline{x}_n$, and an $\varepsilon$-feasible function $f(x_1, \ldots, x_n)$ with rational coefficients,*

*COMPUTE, with probability $\ge 1 - p_0$, rational numbers $\underline{r}$ and $\overline{r}$ which are $\varepsilon$-close to, correspondingly, values $\underline{y}$ and $\overline{y}$ for which*

$$1 - \overline{\delta} \le \frac{\left|\left\{x : x \in \mathbf{x} \,\&\, f(x) \ge \underline{y}\right\}\right|}{|\{x : x \in \mathbf{x}\}|} \le 1 - \underline{\delta} \tag{15}$$

*and*

$$1 - \overline{\delta} \le \frac{|\{x : x \in \mathbf{x} \,\&\, f(x) \le \overline{y}\}|}{|\{x : x \in \mathbf{x}\}|} \le 1 - \underline{\delta}. \tag{16}$$

**Proposition.** *For each tuple $(\varepsilon, \underline{\delta}, \overline{\delta}, p_0)$, there exists a feasible algorithm that solves the corresponding realistic fuzzy computations problem.*

**Proof.** The desired algorithm uses the standard random number generator algorithm that generates numbers $\xi$ uniformly distributed on the interval $[0, 1]$. For each interval $[\underline{x}_i, \overline{x}_i]$, by using such a random number generator, we can compute the value $x_i = \underline{x}_i + \xi \cdot (\overline{x}_i - \underline{x}_i)$ which is uniformly distributed on the interval $[\underline{x}_i, \overline{x}_i]$. If we repeat this procedure $n$ times, for $n$ intervals $\mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$, then we get a tuple $(x_1, \ldots, x_n)$ which is uniformly distributed on the box $\mathbf{x}_1 \times \ldots \times \mathbf{x}_n$.

Now, we can formulate the resulting algorithm:

- first, we select an appropriate natural number $N$, and compute the values
$$\widetilde{\delta} \stackrel{\text{def}}{=} \frac{\underline{\delta} + \overline{\delta}}{2}, \quad \Delta \stackrel{\text{def}}{=} \frac{\overline{\delta} - \underline{\delta}}{2}, \text{ and } v = \lfloor N \cdot \widetilde{\delta} \rfloor; \tag{17}$$
- then, $N$ times we use the above procedure for generating tuples uniformly distributed on the box $\mathbf{x}_1 \times \ldots \times \mathbf{x}_n$, and get $N$ tuples $x^{(1)}, \ldots, x^{(N)}$;
- we apply the given feasible algorithm $f$ to each of these tuples, generating $N$ values $y^{(k)}$ which are $\varepsilon$-close to $f\left(x^{(k)}\right)$:
$$\left|y^{(k)} - f\left(x^{(k)}\right)\right| \le \varepsilon; \tag{18}$$

- we sort these $N$ values $y^{(k)}$ into an increasing sequence

$$y_{(1)} \le y_{(2)} \le \ldots \le y_{(N)}; \tag{19}$$

- finally, we take $\underline{r} = y_{(v)}$ and $\overline{r} = y_{(N-v)}$.

Let us show that for an appropriate value $N$, this Monte-Carlo-type algorithm indeed solves the corresponding realistic fuzzy computation problem.

Indeed, each value $y^{(k)}$ is $\varepsilon$-close to the corresponding value $f\left(x^{(k)}\right)$. Thus, if we sort the values $f\left(x^{(k)}\right)$ into an increasing sequence $f_{(1)} \le f_{(2)} \le \ldots \le f_{(N)}$, then we get $\left|y_{(k)} - f_{(k)}\right| \le \varepsilon$ for all $k$. In particular, we have

$$\left|y_{(v)} - f_{(v)}\right| \le \varepsilon \text{ and } \left|y_{(N-v)} - f_{(N-v)}\right| \le \varepsilon. \tag{20}$$

We will show that the desired inequalities hold for $\underline{y} = f_{(v)}$ and $\overline{y} = f_{(N-v)}$.

For each $y$, let us denote the probability that $f(x) \le y$ under uniform distribution – i.e., the ratio

$$\frac{|\{x : x \in \mathbf{x} \,\&\, f(x) \le y\}|}{|\{x : x \in \mathbf{x}\}|}, \tag{21}$$

by $p$. Then, among $N$ randomly selecting tuples $x$, the mean $E = E[V]$ of number $V$ of tuples for which $f(x) \le y$ is equal to $N \cdot p$. The number $V$ is equal to the sum of $N$ independent identically distributed variables $\eta_1, \ldots, \eta_N$ each of which is equal:

- to 1 if $f(x) \le y$ and
- to 0 if $f(x) > y$.

For each such variable $\eta$, the variance is equal to

$$E\left[(\eta - E[\eta])^2\right] = E\left[(\eta - p)^2\right] =$$

$$p \cdot (1-p)^2 + (1-p) \cdot (0-p)^2 = p \cdot (1-p). \tag{22}$$

Thus, the variance of $V$ is equal to $N \cdot p \cdot (1-p)$, and thus, the standard deviation is equal to $\sigma[V] = \sqrt{N \cdot p \cdot (1-p)}$.

For large $N$, due to the Central Limit Theorem (see, e.g., [21]), the distribution of $V$ is close to Gaussian. Thus, for an appropriate $k_0$ (whose value depend on $p_0$ [21])), with probability $\ge 1 - p_0$, the actual value of $V$ is within the interval

$$[E[V] - k_0 \cdot \sigma[V], E[V] + k_0 \cdot \sigma[V]]. \tag{23}$$

For example:

- for $p_0 = 0.05$, we can take $k_0 = 2$;
- for $p_0 = 0.001$, we can take $k_0 = 3$, and
- for $p_0 = 10^{-8}$, we can take $k_0 = 6$.

With the probability $\ge 1 - p_0$, the difference between the actual value $V$ and the value $E[V] = p \cdot N$ does not exceed

$$k_0 \cdot \sqrt{N} \cdot \sqrt{p \cdot (1-p)}. \tag{24}$$

The largest possible value of the product $p \cdot (1-p)$ is attained when $p = 0.5$; in this case, $\sqrt{p \cdot (1-p)} = 0.5$. Thus, we always have $|V - p \cdot N| \le 0.5 k_0 \cdot \sqrt{N}$. Hence,

$$\left|p - \frac{V}{N}\right| \le \frac{0.5 k_0}{\sqrt{N}}. \tag{25}$$

For $\underline{y} = f_{(v)}$, we have $V = v$ and thus, the actual probability $p$ that $f(x)$ is smaller than or equal to this value $\underline{y}$ satisfies the inequality

$$\left|p - \frac{v}{N}\right| \le \frac{0.5 k_0}{\sqrt{N}}. \tag{26}$$

Here, by definition of $v$, we have $\left|v - \widetilde{\delta} \cdot N\right| \le 1$, hence $\left|\frac{v}{N} - \widetilde{\delta}\right| \le \frac{1}{N}$, thus

$$|p - \widetilde{\delta}| \le \left|p - \frac{v}{N}\right| + \left|\frac{v}{N} - \widetilde{\delta}\right| \le \frac{0.5 k_0}{\sqrt{N}} + \frac{1}{N}. \tag{27}$$

We want to make sure that $p$ is in the interval $\left[\underline{\delta}, \overline{\delta}\right]$. This is equivalent to requiring that $p$ differs from the center $\widetilde{\delta}$ by no more than its half-width $\Delta$, i.e., that $\left|p - \widetilde{\delta}\right| \le \Delta$. Thus, to make sure that our algorithm solves the corresponding problem, it is sufficient to find the value $N$ for which

$$\frac{0.5 k_0}{\sqrt{N}} + \frac{1}{N} \le \Delta. \tag{28}$$

Such a value $N$ is easy to find. For this $N$, the selected value $\underline{y}$ satisfies the first of the two desired inequalities.

Similarly, we can prove that for this same $N$, the selected value $\overline{y} = f_{(N-v)}$ satisfies the second of these inequalities. The proposition is proven.

*Comment* 1. Our preliminary results show that this algorithm is not just theoretically feasible: it indeed produces the desired result in reasonable time. We hope that after this paper appears, practitioners will apply our algorithm to practical problems and thus, test its efficiency of more complex practical examples.

*Comment* 2. The appearance of Monte-Carlo techniques in a paper about fuzzy computation may seem to contradict the fact that many papers on fuzzy computations emphasize that their results are faster to compute than more traditional Monte-Carlo-based techniques. However, there is no contradiction; indeed:

- the computation time of usual fuzzy computation algorithms grows with $n$, while
- the number of iterations $N$ needs for a Monte-Carlo algorithm does not depend on $n$ at all – and the above proof is an example of this fact.

As a result, for large $n$, when the computation time of the usual fuzzy computation algorithms grows very large, Monte-Carlo-type methods become much more efficient. But, on the other hand, when the number of inputs $n$ is reasonably small, the usual methods are much faster – which is exactly what many papers have claimed.

In this paper, our main objective is to find a feasible algorithm, i.e., an algorithm that works reasonably well even for large $n$. This is exactly why we produced the above algorithm. By no means is this algorithm optimal – especially for small $n$, when clearly the usual methods are faster.

*Comment* 3. In this paper, we provide a feasible algorithm for a realistic formulation of maximization and minimization problems. However, similar ideas can be applied to come up with feasible algorithms for solving more complex problems, such as minimax or maximin, where we need to compute

$$\min_{x\in X}\max_{y\in Y} f(x,y) \text{ or } \max_{x\in X}\min_{y\in Y} f(x,y); \tag{29}$$

this is important, e.g., in game theory; see, e.g., [9]. Such problems are also, in general, NP-hard, but if we replace maximum and minimum over all possible value to maximum and minimum over almost all values, then we will be able to produce similar feasible algorithms. Same idea can be applied to more general problems, when we have more than two combinations of minimum and maximum.

These problems appear, e.g., if we start with a general first-order formula – i.e., a formula obtained from equalities and inequalities by applying logical connectives "and", "or", and "not" and quantifiers $\exists x$ and $\forall x$ running over appropriate intervals. Indeed:

- each strict inequality $f > g$ can be approximated, with any possible accuracy $\delta > 0$, as a non-strict inequality
$$f \geq g + \delta; \tag{30}$$
- each equality $f = g$ can be represented as two inequalities $f \geq g$ and $g \geq f$;
- each inequality $f \geq g$ can be represented as $h \geq 0$, for
$$h \stackrel{\text{def}}{=} f - g; \tag{31}$$
- the formula $f \geq 0 \,\&\, g \geq 0$ can be represented as
$$\min(f,g) \geq 0; \tag{32}$$
- the formula $f \geq 0 \vee g \geq 0$ can be represented as
$$\max(f,g) \geq 0; \tag{33}$$
- the formula $\exists x\,(f \geq 0)$ is equivalent to $\max\limits_x f \geq 0$; and
- the formula $\forall x\,(f \geq 0)$ is equivalent to $\min\limits_x f \geq 0$.

## Acknowledgments

The authors are greatly thankful to the anonymous referees for valuable suggestions.

## References

[1] O. Atli and C. Kahraman, "Resource-constrained project scheduling problem with multiple execution modes and fuzzy/crisp activity durations", *Journal of Intelligent & Fuzzy Systems*, 2014, Vol. 26, No. 4, pp. 2001–2020.

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

[3] L. O. Fuentes, *Applying Uncertainty Formalisms to Well-Defined Problems: Experimental and Theoretical Foundations*, Master's Thesis, University of Texas at El Paso, Department of Computer Science, 1991.

[4] M. Hidalgo-Herrero, P. Rabanal, I. Rodríguez, and F. Rubio, "Comparing problem solving strategies for NP-hard optimization problems", *Fundamenta Informaticae*, 2013, Vol. 124, No. 1–2, pp. 1–25.

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

[6] V. Kreinovich, "S. Maslov's iterative method: 15 years later (freedom of choice, neural networks, numerical optimization, uncertainty reasoning, and chemical computing)", In: V. Kreinovich and G. Mints (eds.), *Problems of Reducing the Exhaustive Search*, American Mathematical Society, Providence, RI, 1997, pp. 175–189.

[7] V. Kreinovich, A. Lakeyev, J. Rohn, and P. Kahl, *Computational Complexity and Feasibility of Data Processing and Interval Computations*, Kluwer, Dordrecht, 1998.

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

[9] R. B. Myerson, *Game Theory: Analysis of Conflict*, Harvard University Press, Harvard, Massachusetts, 1997.

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

[11] N. Nilsson, *Artificial Intelligence: A New Synthesis*, Morgan Kauffman, San Francisco, California, 2003.

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

[13] Wei Pang, Kang-Ping Wang, Chun-Guang Zhou, and Long-Jiang Dong, "Fuzzy discrete particle swarm optimization for solving traveling salesman problem", *Proceedings of the Fourth International Conference on Computer and Information Technology CIT'04*, Wuhan, China, September 16, 2004, pp. 796–800.

[14] C. H. Papadimitriou, *Computational Complexity*, Addison-Wesley, San Diego, 1994.

[15] F. Rajabi, S. E. Najafi, M. Hajiaghaei-Keshteli, and S. Molla-Alizadeh-Zavardehi, "Solving fuzzy step fixed charge transportation problems via metaheuristics", *International Journal of Research in Industrial Engineering*, 2013, Vol. 2, No. 3, pp. 14–24.

[16] E. D. Rodriguez Velasquez, C. M. Chang Albitres, and V. Kreinovch, "Fuzzy ideas explain a complex heuristic algorithm for gauging pavement conditions", *Mathematical Structures and Modeling*, 2018, Vol. 47, pp. 82–90.

[17] U. Roy and Xiaoyan Zhang, "A heuristic approach to n/m job shop scheduling: fuzzy dynamic scheduling algorithms", *Production Planning & Control*, 1996, Vol. 7, No. 3, pp. 299–311.

[18] E. D. Rodriguez Velasquez, C. M. Chang Albitres, and V. Kreinovch, "Measurement-type 'calibration' of expert estimates improves their accuracy and their usability: pavement engineering case", *Proceedings of the IEEE Symposium on Computational Intelligence for Engineering Solutions CIES'2018*, Bengaluru, India, November 18–21, 2018.

[19] E. D. Rodriguez Velasquez, C. M. Chang Albitres, Thach Ngoc Nguyen, O. Kosheleva, and V. Kreinovich, "How to take expert uncertainty into account: economic approach illustrated by pavement engineering applications", In: V. Kreinovich and S. Sriboonchitta (eds.), *Structural Changes and Their Econometric Modeling*, Springer Verlag, Cham, Switzerland, 2019.

[20] J. Saray, Y. Badr, and A. Abraham, "An enhanced fuzzy-genetic algorithm to solve satisfiability problems", *Proceedings of the International Conference on Computer Modelling and Simulation UKSim'11*, Cambridge, UK, March 25–27, 2009.

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

[22] S. A. Vavasis, *Nonlinear Optimization: Complexity Issues*, Oxford University Press, New York, 1991.

[23] M. Verma and K. K. Shukla, "Application of fuzzy optimization to the orienteering problem", *Advances in Fuzzy Systems*, 2015, Vol. 2015, Paper 569248.

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