Data Processing under Fuzzy Uncertainty: Towards More Efficient Algorithms

Hung T. Nguyen[1], Olga Kosheleva[2a], and Vladik Kreinovich[2b]

[1]Department of Mathematical Sciences, New Mexico State University
Las Cruces, New Mexico, USA, and
Faculty of Economics, Chiang Mai University, Chiang Mai, Thailand
hunguyen@nmsu.edu
[2]Departments of [a]Teacher Education and [b]Computer Science
University of Texas at El Paso, El Paso, Texas 79968, USA
[a]olgak@utep.edu, [b]vladik@utep.edu

1. Outline

- When we process data:
  - i.e., when we apply a data processing algorithm $y = f(x_1, \ldots, x_n)$ to several inputs $x_1, \ldots, x_n$
  - often, the only information about some of the inputs x_i comes from experts,
  - and experts describe this information by using imprecise ("fuzzy") words from natural language such as "small".
- In this case, a natural idea is to use fuzzy methodology to transform this information into membership functions $\mu_i(x_i)$.
- The ultimate objective of data processing is to provide information about the quantity y.
- Thus, we need to compute the membership f-n corresponding to y.
- This membership function can be determined by applying Zadeh's extension principle.

2. Outline (cont-d)

- The existing algorithms for computing the desired membership function for y are frequently time-consuming.
- In this talk, we describe new faster algorithms for this computation.

3. Need for Data Processing

- To understand the current state of the world, we need to know the values of different quantities that describe this state; e.g.:
  - to understand the current weather at some location,
  - we need to know the temperature, humidity, wind speed and wind direction, and the current amount of rain or snow.
- All these quantities can be directly measured – or, if needed, estimated.
- E.g., by going out, we can feel the temperature with some reasonable accuracy.
- In other practical situations, we are interested in a quantity y that is difficult (or even impossible) to measure or estimate directly.
- For example, it is difficult to directly measure or estimate:
  - the amount of oil in an oil field, or
  - the distance to a faraway star.

4. Need for Data Processing (cont-d)

- In many such cases, we know a relation $y = f(x_1, \ldots, x_n)$ between:
  - the desired quantity y and
  - auxiliary quantities $x_1, \ldots, x_n$ which are easier to measure and/or to estimate.
- For example, to estimate the amount of oil, we can:
  - perform some seismic measurements, and then
  - use the results of these measurements to estimate the geological structure of this oil field and
  - thus, to estimate the amount of oil in this field.
- To estimate the distance to a faraway star, we:
  - observe its location at two different seasons, when the Earth is on the opposite sides of the Sun, and
  - use trigonometry – and the known diameter of the Earth orbit – to estimate the desired distance.

5. Exact vs. Approximate Algorithms

- In some cases – e.g., for estimating the distance to a faraway star:
  - the known function $y = f(x_1, \ldots, x_n)$
  - provides the exact relation between the corresponding quantities.
- In other cases – e.g., for estimating the amount of oil – the known algorithms provide only an approximate estimate.

6. Need to Take Uncertainty into Account

- In situations when we know the exact dependence $y = f(x_1, \ldots, x_n)$ between x_i and y:
  - once we know the actual values x_i^{act} of the auxiliary quantities

$$x_1, \ldots, x_n,$$

  - we can compute the actual value y^{act} of the desired quantity y as

$$y_i^{\text{act}} = f\left(x_i^{\text{act}}, \ldots, x_n^{\text{act}}\right).$$

- However:
  - whether we get the values of the quantities x_i from measurements or from expert estimates,
  - we rarely know the exact values of these quantities.
- Measurements are usually more accurate than expert estimates, but they are never absolutely accurate.

7. Need to Take Uncertainty into Account (cont-d)

- The measurement result $\widetilde{x}_i$ is, in general, somewhat different from the actual (unknown) value $x_i^{\rm act}$ of the corresponding quantity.
- The corresponding difference $\Delta x_i \stackrel{\rm def}{=} \widetilde{x}_i - x_i^{\rm act}$ is known as the *measurement error.*
- In many practical situations, the only information that we have about the measurement error Δ_i is the upper bound on its absolute value:

$$|\Delta x_i| \le \Delta_i.$$

- In this case, after the measurement, the only information that we gain about the actual value $x_i^{\rm act}$ is that this value belongs to the interval

$$[\widetilde{x}_i - \Delta_i, \widetilde{x}_i + \Delta_i].$$

8. Need to Take Uncertainty into Account (cont-d)

- Because of the measurement errors:
  - when we apply the algorithm $y = f(x_1, \ldots, x_n)$ to the measurement results $\widetilde{x}_i$,
  - we get the value $\widetilde{y} = f(\widetilde{x}_1, \ldots, \widetilde{x}_n)$ which is, in general, different from the actual value $f\left(x_1^{\text{act}}, \ldots, x_n^{\text{act}}\right)$.
- Similarly, expert estimates are usually approximate.
- So, when we apply the algorithm $y = f(x_1, \ldots, x_n)$ to expert estimates, we get an approximate value of y.
- From the practical viewpoint, it is important to know how accurate is our estimate of the desired quantity y.

9. Need to Take Uncertainty into Account (cont-d)

- For example, if we estimate that the amount of oil in a given oil field is approximately 100 million tons, then:
  - it could be 100 ± 10 – in which case we need to start exploiting it, or
  - it could be 100 ± 200 – in which case we are not even sure that there is any oil, so a further analysis is necessary.

10. Fuzzy Uncertainty: Case of Exactly Known Dependence

- Experts often describe their estimates by using imprecise ("fuzzy") words from natural language.
- Example: "small" or "much smaller than 100", etc.
- To describe such imprecise knowledge in precise computer-understandable terms, a natural idea is to use *fuzzy methodology*.
- This methodology was specifically designed to provide such descriptions.
- In this methodology, to describe the expert's statement about a quantity x_i, we ask the expert to estimate:
  - for each possible value X_i of this quantity,
  - the degree $\mu_i(X_i)$ from the interval $[0,1]$ to which, in his/her opinion, this value is consistent with this statement.

11. Case of Exactly Known Dependence (cont-d)

- The resulting function $\mu(X_i)$ is known as the *membership function*, or, alternatively, as the *fuzzy set*;
  - the case when the information about some quantity x_i comes from measurement –
  - and in which we thus know the interval $[\widetilde{x}_i - \Delta_i, \widetilde{x}_i + \Delta_i]$ of possible values of x_i,
  - can be viewed as a particular case of the fuzzy case, when:
    - $\mu_i(X_i) = 1$ for all X_i from the given interval, and
    - $\mu_i(X_i) = 0$ for all values X_i which are outside this interval.
- So:
  - in this case, when we know the exact dependence between y and x_i, and the inputs are known with fuzzy uncertainty,
  - we arrive at the following problem:

12. Case of Exactly Known Dependence (cont-d)

- *We know:*
  - the membership functions $\mu_i(X_i)$ describing our knowledge about the quantities $x_1, \ldots, x_n$, and
  - the dependence $y = f(x_1, \ldots, x_n)$ between x_i and y.
- *We want to describe*: the resulting information about y.
- Since our information about the inputs is imprecise, the resulting information about y is also imprecise.
- So, it should be also described by a membership function $\mu(Y)$.

13. Fuzzy Uncertainty: General Case

- In the previous slide, we assumed that the formula $y = f(x_1, \ldots, x_n)$ describe the exact relation between y and x_i.
- However, often, the function $f(x_1, \ldots, x_n)$ provides only an *approximate* description of this dependence: $y \approx f(x_1, \ldots, x_n)$.
- In many such situations, we only have expert opinion about the inaccuracy $m \stackrel{\text{def}}{=} y - f(x_1, \ldots, x_n)$.
- E.g., we know that this difference is small, or that it is cannot be much larger than 10, etc.;
  - to describe this imprecise natural-language information about the difference m in precise computer-understandable terms,
  - a natural idea is to use the general fuzzy methodology.
- This methodology enables us to transform expert opinion into a membership function $\mu_M(m)$.

14. Fuzzy Uncertainty: General Case (cont-d)

- In this case, we know that $y = f(x_1, \ldots, x_n) + m$, and we know the membership functions corresponding to $x_1, \ldots, x_n$, and to m.
- From the mathematical viewpoint, we can view m as an additional input; we will denote it by x_{n+1}.
- So, from this mathematical viewpoint, we know the exact dependence of y on the $n+1$ inputs: $y = F(x_1, \ldots, x_n, x_{n+1})$, where we denoted

$$F(x_1, \ldots, x_n, x_{n+1}) \stackrel{\text{def}}{=} f(x_1, \ldots, x_n) + x_{n+1}.$$

- Thus, we get the following equivalent reformulation of the problem:
- *We know:*
  - the membership functions $\mu_i(X_i)$ describing our knowledge about the quantities $x_1, \ldots, x_n, x_{n+1}$, and
  - the dependence $y = F(x_1, \ldots, x_n, x_{n+1})$ between x_i and y.
- *We want to describe*: the resulting information about y.

15. Fuzzy Uncertainty: General Case (cont-d)

- Thus, from the computational viewpoint:
  - this more realistic formulation can be reduced to
  - the previously considered case when we know the exact dependence between x_i and y.
- In view of this reduction, in the following, we will assume that the exact dependence between y and x_i is known.
- Indeed, from the computational viewpoint, the general case can be reduced to this exact-dependence case.

16. How This Problem Is Usually Formalized: Zadeh's Extension Principle

- In fuzzy technique, the information about each input x_i is described by assigning:
  - to each real number X_i,
  - the degree $\mu_i(X_i) \in [0,1]$ to which, according to the expert, this number is a possible value of x_i.
- The corresponding function $\mu_i(X_i)$ is known as a *membership function*.
- Based on this information, we need to find a similar membership function $\mu(Y)$ for the quantity $y = f(x_1, \ldots, x_n)$.
- This function should describe, for each real number Y, the degree to which this number is a possible value of y.

17. Zadeh's Extension Principle (cont-d)

- Intuitively, a number Y is a possible value of the quantity y if and only if there exist values $X_1, \ldots, X_n$:
  - which are possible values of the corresponding inputs and
  - for which $Y = f(X_1, \ldots, X_n)$.
- We know the degrees $\mu_i(X_i)$ to which each X_i is a possible value of x_i; so:
  - we can use the simplest way to describing "and" and "or" in fuzzy techniques – as min and max – and
  - take into account that "there exist" is nothing else by an infinite "or".
- We thus conclude that

$$\mu(Y) = \max\{\min(\mu_1(X_1), \ldots, \mu_n(X_n)) : f(X_1, \ldots, X_n) = Y\}.$$

- This formula was first introduced by Zadeh and is thus known as *Zadeh's extension principle.*

18. Zadeh's Extension Principle (cont-d)

- So, we can now formulate our problem in precise terms:
- *We know:*
  - the membership functions $\mu_i(X_i)$ describing our knowledge about the quantities $x_1, \ldots, x_n$, and
  - the dependence $y = f(x_1, \ldots, x_n)$ between x_i and y.
- *We want*: to estimate the function $\mu(Y)$.

19. State-of-the-Art Algorithms for Data Processing under Fuzzy Uncertainty: Main Idea

- A known way to perform the corresponding computations is to use alpha-cuts, i.e., sets defined:
  - as $\mathbf{x}(\alpha) = \{x : \mu(x) \ge \alpha\}$ for $\alpha > 0$ and
  - as the closure $\mathbf{x}(0) = \overline{\{x : \mu(x) > 0\}}$ for $\alpha = 0$.
- The use of α-cuts is based on the fact that for Zadeh's extension principle, for each $\alpha \in [0, 1]$:
  - the α-cut $\mathbf{y}(\alpha)$ of y is equal to
  - the range of the function $f(x_1, \ldots, x_n)$ on α-cuts $\mathbf{x}_i(\alpha)$ of the inputs x_i:
  $$\mathbf{y}(\alpha) = \{f(x_1, \ldots, x_n) : x_i \in \mathbf{x}_i(\alpha)\}.$$
- Thus, to solve the above problem, we need to compute, for several possible values of α, the corresponding range.

20. α-Cuts Are Usually Intervals

- Usually, the membership functions $\mu_i(X_i)$ first increase and then decrease.
- In this case, all α-cuts are intervals.
- The function $y = f(x_1, \ldots, x_n)$ is usually continuous.
- It is known that the range of a continuous function on a closed connected domain is an interval.
- Thus, the resulting range is also an interval.
- So, for each α, we face the following problem:

21. α-Cuts Are Usually Intervals (cont-d)

- *We know:*
  - the intervals $\mathbf{x}_i(\alpha)$ – α-cuts of the corresponding membership functions $\mu_i(X_i)$, and
  - the dependence $y = f(x_1, \ldots, x_n)$ between x_i and y.
- *We want*: to estimate the endpoints $\underline{y}(\alpha)$ and $\overline{y}(\alpha)$ of the interval $\mathbf{y}(\alpha) = [\underline{y}(\alpha), \overline{y}(\alpha)]$ determined by the formula (2).

22. How Many α-Cuts Do We Need?

- In the traditional application of fuzzy techniques, the membership degrees come from expert estimates.
- We can use the fact that, according to psychologists, intuitively, we classify all the objects into maximum of seven plus two categories.
- I.e., between 5 and 9.
- This means, in particular, that we can have no more than 9 really different degrees.
- Indeed, hardly anyone would say that some statement has a degree of certainty 0.51 as opposed to 0.5.
- In this case, it makes sense to process 9 or so different values α.
- In some applications of fuzzy techniques, we "tune" the original expert estimates by using real data.
- In such cases, we can get more accurate degrees – and thus, we will need to process more than 9 different values α.

23. How Many α-Cuts Do We Need (cont-d)

- In general, let us denote the number of α-levels for which we want to estimate the values $\underline{y}(\alpha)$ and $\overline{y}(\alpha)$ by A.
- In both cases, we have $A \geq 9$.

24. How to Compute the Range: Interval Computations

- For each α, we face the following problem:
- *We know:*
  - the intervals $\mathbf{x}_i = [\underline{x}_i, \overline{x}_i]$, and
  - the dependence $y = f(x_1, \ldots, x_n)$ between x_i and y.
- *We want*: to estimate the endpoints $\underline{y}$ and $\overline{y}$ of the interval

$$[\underline{y}, \overline{y}] = \{f(x_1, \ldots, x_n) : x_i \in \mathbf{x}_i \text{ for all } i\}.$$

- The above problem of computing $\underline{y}$ and $\overline{y}$ is known as the main problem of *interval computation*.
- It is known that, in general, this interval computation problem is NP-hard already for quadratic functions $f(x_1, \ldots, x_n)$.
- This means that for large n, *exact* computation of the range is not always feasible.
- However, there are efficient *approximate* interval techniques.

25. Interval Arithmetic

- Most interval techniques are based on the fact that:
  - for the cases when data processing consists of a single arithmetic operation,
  - we can explicitly compute the range of the resulting value:

$$[\underline{x}_1, \overline{x}_1] + [\underline{x}_2, \overline{x}_2] = [\underline{x}_1 + \underline{x}_2, \overline{x}_1 + \overline{x}_2];$$

$$[\underline{x}_1, \overline{x}_1] - [\underline{x}_2, \overline{x}_2] = [\underline{x}_1 - \overline{x}_2, \overline{x}_1 - \underline{x}_2];$$

$$[\underline{x}_1, \overline{x}_1] \cdot [\underline{x}_2, \overline{x}_2] = [\min(\underline{x}_1 \cdot \underline{x}_2, \underline{x}_1 \cdot \overline{x}_2, \overline{x}_1 \cdot \underline{x}_2, \overline{x}_1 \cdot \overline{x}_2),$$

$$\max(\underline{x}_1 \cdot \underline{x}_2, \underline{x}_1 \cdot \overline{x}_2, \overline{x}_1 \cdot \underline{x}_2, \overline{x}_1 \cdot \overline{x}_2);$$

$$1/[\underline{x}_1, \overline{x}_1] = [1/\overline{x}_1, 1/\underline{x}_1] \text{ if } 0 \notin [\underline{x}_1, \overline{x}_1];$$

$$[\underline{x}_1, \overline{x}_1]/[\underline{x}_2, \overline{x}_2] = [\underline{x}_1, \overline{x}_1] \cdot (1/[\underline{x}_2, \overline{x}_2]).$$

- These formulas are known as formulas of *interval arithmetic*.

26. From Interval Arithmetic to Straightforward ("Naive") Interval Computations

- The next step in designing state-of-the-art interval computation algorithms is straightforward ("naive") interval computations.
- This technique is based on the fact that in a computer, any algorithm is implemented as a sequence of arithmetic operations.
- If we replace each arithmetic operation with the corresponding operation of interval arithmetic, we get an enclosure for the desired range.
- For example, when a computer computes the value of a function $f(x) = x \cdot (1 - x)$, it:
  - first computes the difference $r = 1 - x$, and
  - then computes the product $x \cdot r$.

27. Straightforward Interval Computations (cont-d)

- So, to find an enclosure for the range of this function on the interval $[0, 1]$, we can:
  - first apply interval subtraction to find the range for r as
  $$[1, 1] - [0, 1] = [1 - 1, 1 - 0] = [0, 1],$$
  - and then apply interval multiplication to compute
  $$[0, 1] \cdot [0, 1] =$$
  $$[\min(0 \cdot 0, 0 \cdot 1, 1 \cdot 0, 1 \cdot 1), \max(0 \cdot 0, 0 \cdot 1, 1 \cdot 0, 1 \cdot 1)] = [0, 1].$$
- The resulting range $[0, 1]$ is clearly an enclosure for the actual range $[0, 0.25]$, but a very crude one.

28. State-of-the-Art Interval Computation Technique – General Case: Main Idea

- State-of-the-art interval computation packages:
  - compute much narrower (and thus, more practically useful) enclosures
  - than straightforward interval computations.
- One of the main ideas is to use *centered form.*
- Let us explain how this technique works.
- In this technique, on each input interval $[\underline{x}_i, \overline{x}_i]$, we select a representative value $\widetilde{x}_i$.
- This could be a midpoint, this could be a different point from this interval.
- Then, each possible value $x_i \in [\underline{x}_i, \overline{x}_i]$ can be represented as $\widetilde{x}_i - \Delta x_i$, where $\Delta x_i \in [\Delta_i^-, \Delta_i^+] \stackrel{\text{def}}{=} [\widetilde{x}_i - \overline{x}_i, \widetilde{x}_i - \underline{x}_i]$.

29. State-of-the-Art Interval Computation (cont-d)

- The centered form technique is based on the Intermediate Value Theorem:
  - for each combination of values $x_i \in [\underline{x}_i, \overline{x}_i]$,
  - there exist values ξ_{ij} from the same intervals $[\underline{x}_i, \overline{x}_i]$ for which

$$\Delta y = f(\widetilde{x}_1, \ldots, \widetilde{x}_n) - f(\widetilde{x}_1 - \Delta x_1, \ldots, \widetilde{x}_n - \Delta x_n) =$$

$$\sum_{i=1}^{n} \frac{\partial f}{\partial x_i}_{|x_j = \xi_{ij},} \cdot \Delta x_i.$$

- We know that, since $\xi_{ij} \in [\underline{x}_i, \overline{x}_i]$, each partial derivative value belongs to the r ange of this partial derivative on these intervals.
- Thus, it belongs the enclosure $\mathbf{D}_i([\underline{x}_1, \overline{x}_1], \ldots, [\underline{x}_n, \overline{x}_n])$ of this range.
- This enclosure can be computed, e.g., by straightforward interval computations.

30. State-of-the-Art Interval Computation (cont-d)

- Also, we know that $\Delta x_i \in [\Delta_i^-, \Delta_i^+]$.
- Thus, $\Delta y \in \mathbf{D} \stackrel{\text{def}}{=} \sum_{i=1}^n \mathbf{D}_i([\underline{x}_1, \overline{x}_1], \ldots, [\underline{x}_n, \overline{x}_n]) \cdot [\Delta_i^-, \Delta_i^+]$.
- Hence, $y \in \mathbf{Y} \stackrel{\text{def}}{=} \widetilde{y} - \mathbf{D}$.
- The right-hand side of this formula is what is called the *centered form*.

31. State-of-the-Art Interval Computation Technique – General Case: Centered-Form Algorithm

- We are given n intervals $[\underline{x}_i, \overline{x}_i]$ $(1 \le i \le n)$ and a function

$$f(x_1, \ldots, x_n).$$

- To compute an enclosure for the desired range, we do the following:
- On each interval, we select a point $\widetilde{x}_i$.
- We compute $\Delta^- = \widetilde{x}_i - \overline{x}_i$, $\Delta^+ = \widetilde{x}_i - \underline{x}_i$, and $\widetilde{y} = f(\widetilde{x}_1, \ldots, \widetilde{x}_n)$.
- For each i, we use straightforward interval computations to compute an enclosure $\mathbf{D}_i([\underline{x}_1, \overline{x}_1], \ldots, [\underline{x}_n, \overline{x}_n])$ of the i-th partial derivatives.
- Then, we use the above formulas to compute the enclosure $\mathbf{Y}$.
- In the fuzzy case, we perform such computations for all $A \ge 9$ selected value α.

32. The Centered Form Algorithm Is Asymptotically Optimal

- It is known that the centered form algorithm is asymptotically the most accurate, in the following sense:
  - for some constant C, it provides the $C \cdot h^2$ accuracy in estimating the range, where h is largest width of the input intervals, while
  - estimating the range with higher accuracy $c \cdot h^2$ is NP-hard for sufficiently small c.

33. Linearization Case

- In many practical situations, the estimation error is relatively small.
- In other words, the interval width $\overline{x}_i - \underline{x}_i$ is much smaller than the actual values x_i from this interval.
- It is, e.g., 10% or 20% of this value.
- Thus, the difference $\Delta x_i \stackrel{\text{def}}{=} \widetilde{x}_i - x_i$ between the midpoint $\widetilde{x}_i \stackrel{\text{def}}{=} (\underline{x}_i + \overline{x}_i)/2$ and a possible value $x_i \in [\underline{x}_i, \overline{x}_i]$ is also small.
- This difference is bounded by the interval's half-width $\Delta_i \stackrel{\text{def}}{=} (\overline{x}_i - \underline{x}_i)/2$, so $\Delta x_i \in [-\Delta_i, \Delta_i]$.
- In this case, terms quadratic in such small differences are much smaller than linear terms.
- E.g., square of 10% is 1% which is much smaller than 10%.

34. Linearization Case (cont-d)

- Thus, we can *linearize* the problem, i.e.:
  - expand the difference $\widetilde{y} - y$, where $\widetilde{y} \stackrel{\text{def}}{=} f(\widetilde{x}_1, \ldots, \widetilde{x}_n)$ is the result of processing midpoints, in Taylor series in terms of Δx_i, and
  - keep only linear terms in this expansion.
- As a result, we get:

$$\Delta y = y - \widetilde{y} = f(\widetilde{x}_1 - \Delta x_1, \ldots, \widetilde{x}_n - \Delta x_n) - f(\widetilde{x}_1, \ldots, \widetilde{x}_n) \approx$$

$$\sum_{i=1}^{n} c_i \cdot \Delta x_i, \text{ where } c_i = \frac{\partial f}{\partial x_i}.$$

- For a linear function Δy, its largest possible value is attained when Δx_i attains:
  - its largest possible value Δ_i when $c_i \geq 0$ and
  - its smallest possible value $-\Delta_i$ when $c_i \leq 0$.

35. Linearization Case (cont-d)

- The resulting largest value Δ of the difference $\Delta y \stackrel{\text{def}}{=} \widetilde{y} - y$ is equal to $\Delta = \sum\limits_{i=1}^{n} |c_i| \cdot \Delta_i$.
- Similarly, one can show that the smallest possible value of Δy is $-\Delta$.
- So, the resulting interval range of possible values of y is $[\widetilde{y}-\Delta, \widetilde{y}+\Delta]$.
- To use the above formula for Δ, we need to know the values of all the partial derivatives c_i.
- For small n, we can feasibly compute all these values by the usual numerical differentiation techniques, e.g., as

$$c_i \approx \frac{f(\widetilde{x}_1, \ldots, \widetilde{x}_{i-1}, \widetilde{x}_i + h_i, \widetilde{x}_{i+1}, \ldots, \widetilde{x}_n) - \widetilde{y}}{h_i}.$$

- Here, h_i are some small values.
- For $h_i = \Delta_i$, we thus arrive at the first of the following two algorithms.

36. State-of-the-Art Techniques for the Linearization Case

- We know $\widetilde{x}_i$, Δ_i, and $f(x_1, \ldots, x_n)$.
- Based on this information, we compute $\widetilde{y} = f(\widetilde{x}_1, \ldots, \widetilde{x}_n)$, then compute

$$\Delta = \sum_{i=1}^{n} |f(\widetilde{x}_1, \ldots, \widetilde{x}_{i-1}, \widetilde{x}_i + \Delta_i, \widetilde{x}_{i+1}, \ldots, \widetilde{x}_n) - \widetilde{y}|.$$

- Then, $[\underline{y}, \overline{y}] = [\widetilde{y} - \Delta, \widetilde{y} + \Delta]$.
- An alternative algorithm – which for large n, requires fewer than n calls to the program for computing f – is motivated by the fact that:
  - for complex data processing algorithms,
  - we often have thousands of inputs,
  - so computing all partial derivatives would take too long.

37. State-of-the-Art Techniques for the Linearization Case (cont-d)

- In this case, it is possible to use Cauchy deviate Monte-Carlo method, which is based on the fact that:
  - if the variables Δx_i are Cauchy distributed with parameters Δ_i,
  - then the linear combination $\Delta y = \sum c_i \cdot \Delta x_i$ is also Cauchy distributed, with parameter $\Delta = \sum |c_i| \cdot \Delta_i$.
- Here – in contrast to numerical differentiation – the number of calls to the algorithm f does not depend on the number of variables n.
- This number of calls depends only on the desired accuracy and remains constant when n grows.

38. Main Limitation of State-of-the-Art Algorithms: They Sometimes Require Too Much Computation Time

- The state-of-the-art approach to data processing under interval uncertainty means computing the range for $A \geq 9$ different values α.
- Each such computation-of-the-range involves using the data processing algorithm $f(x_1, \ldots, x_n)$ at least once.
- Many data processing algorithms are very complex.
- For complex algorithms $f(x_1, \ldots, x_n)$, each range computation is already time-consuming.
- The need to repeat this computation $A \geq 9$ times increases the computation time by an order of magnitude.
- It can, thus, make the computations not practical.

39. State-of-the-Art Algorithms Require Too Much Time (cont-d)

- For example, an accurate prediction of tomorrow's weather may takes several hours on a high-performance computer; so:
  - if we need to repeat these computations $A \geq 9$ times to find a good description of the accuracy of the prediction results,
  - this will take more than a day.
- This makes no sense, since by then we will already observe tomorrow's weather.
- It is therefore desirable to speed up computations.

40. New Faster Algorithm for Data Processing under Fuzzy Uncertainty: Main Idea

- We consider the problem of data processing under fuzzy uncertainty.
- The state-of-the-art centered form algorithm includes computing the enclosures for the partial derivatives for all $A \geq 9$ values α.
- Computing each of these enclosures is the main time-consuming step of this algorithm.
- Everything else is a few additions and multiplications.
- To speed up, we propose to use the fact that all α-cuts are all subsets of the α-cut corresponding to $\alpha = 0$.
- Thus, all the values of the partial derivative are contained in the enclosure corresponding to $\alpha = 0$.
- So, we can compute such enclosures only once.

41. General Case: New Algorithm

- We select value $\widetilde{x}_i \in [\underline{x}_i(1), \overline{x}_i(1)]$ and compute $\widetilde{y} = f(\widetilde{x}_1, \ldots, \widetilde{x}_n)$.
- For each α, we compute $\Delta_i^-(\alpha) = \widetilde{x}_i - \overline{x}_i(\alpha)$, $\Delta_i^+(\alpha) = \widetilde{x}_i - \underline{x}_i(\alpha)$, and

$$\mathbf{D}(\alpha) = \sum_{i=1}^{n} \mathbf{D}_i([\underline{x}_1(0), \overline{x}_1(0)], \ldots, [\underline{x}_n(0), \overline{x}_n(0)]) \cdot [\Delta_i^-(\alpha), \Delta_i^+(\alpha)].$$

- Finally, we estimate $\mathbf{y}(\alpha)$ as $\widetilde{y} - \mathbf{D}(\alpha)$.
- The resulting estimate is still asymptotically optimal.
- However, it requires only *one* estimation of the ranges of the derivatives.
- It is, thus, $A \geq 9$ *times faster than the above state-of-the-art α-by-α algorithm.*

42. Linearized Case: Important Subcase

- Often, all membership functions are "of the same type": e.g.:
  - all are symmetric triangular, or
  - all are Gaussian.
- In precise terms, for each of these two families:
  - all the membership functions $\mu_i(X_i)$ from a family are obtained from some standard membership function $\mu_0(X)$
  - by some scaling $X \mapsto s \cdot X$ (with $s > 0$) and shift $X \mapsto X + c$, i.e., $\mu_i(X_i) = \mu_0(s_i \cdot X_i + c_i)$ for some s_i and c_i.
- For each membership function, there is some "most probable" value.
- E.g., the midpoint of the 1-cut, i.e., of the set of all the values for which the degree of possibility is 1:
  - if for the original membership function this value is not 0,
  - we can appropriately shift this standard function, and get a new standard function for which this point is 0.

43. Linearized Case: Important Subcase (cont-d)

- Shifting the standard membership function does not change the class of all membership functions obtained from it by shifts and scalings.
- Thus, without losing generality, we can safely assume that for the standard function, the "most probable" value is 0.
- In this case, the α-cuts for x_i are determined by the α-cuts $[\ell(\alpha), r(\alpha)]$ of the standard membership function $\mu_0(X)$.
- Indeed, here, $\mu_i(X_i) \geq \alpha$ is equivalent to $\mu_0(s_i \cdot X_i + c_i) \geq \alpha$, i.e., to $s_i \cdot X_i + c_i \in [\ell(\alpha), r(\alpha)]$, or, equivalently, to

$$\ell(\alpha) \leq s_i \cdot X_i + c_i \leq r(\alpha).$$

- Subtracting c_i from all sides of this inequality and dividing all sides by $s_i > 0$, we conclude that $\mu_i(X_i) \geq \alpha$ is equivalent to

$$(1/s_i) \cdot \ell(\alpha) + (c_i/s_i) \leq X_i \leq (1/s_i) \cdot r(\alpha) + (c_i/s_i).$$

$$\text{i.e., to } a_i \cdot \ell(\alpha) + b_i \leq X_i \leq a_i \cdot \ell(\alpha) + b_i.$$

44. Linearized Case: Important Subcase (cont-d)

- Here we denoted $a_i \stackrel{\text{def}}{=} 1/s_i$ and $b_i \stackrel{\text{def}}{=} c_i/s_i$.
- Thus, the α-cut $\mathbf{x}_i(\alpha)$ of the i-th input has the form

$$\mathbf{x}_i(\alpha) = [a_i \cdot \ell(\alpha) + b_i, a_i \cdot r(\ell) + b_i].$$

- In particular, for each of these functions, the most probable value is

$$a_i \cdot 0 + b_i = b_i.$$

- We will show that in this case, we can also drastically speed up computations.

45. Linearized Case – Important Subcase: Analysis of the Problem

- In the linearized case:
  - once we know the value $\widetilde{y} = f(b_1, \ldots, b_n)$ corresponding to the most probable values b_i,
  - for the difference $\Delta y \stackrel{\text{def}}{=} \widetilde{y} - f(x_1, \ldots, x_n)$, we get the expression $\Delta y = \sum\limits_{i=1}^{n} c_i \cdot \Delta x_i$, where $\Delta x_i \stackrel{\text{def}}{=} b_i - x_i$.
- When $x_i \in [a_i \cdot \ell(\alpha) + b_i, a_i \cdot r(\ell) + b_i]$, then the difference $\Delta x_i = b_i - x_i$ belongs to the interval $[\Delta_i^-(\alpha), \Delta_i^+(\alpha)] = [-a_i \cdot r(\alpha), -a_i \cdot \ell(\alpha)]$.
- Similarly to the above derivation of the linearization case:
  - to compute the range $[\Delta^-(\alpha), \Delta^+(\alpha)]$ of the linear function $\sum c_i \cdot \Delta x_i$ when $\Delta x_i \in [\Delta_i^-(\alpha), \Delta_i^+(\alpha)]$,
  - we can represent each input interval as

$$\left[\widetilde{\Delta}_i(\alpha) - \Delta_i(\alpha), \widetilde{\Delta}_i(\alpha) + \Delta_i(\alpha)\right].$$

46. Linearized Case – Important Subcase: Analysis of the Problem (cont-d)

- Here, $\widetilde{\Delta}_i(\alpha)=\dfrac{\Delta_i^-(\alpha)+\Delta_i^+(\alpha)}{2}=-a_i\cdot\dfrac{\ell(\alpha)+r(\alpha)}{2}$, and
$$\Delta_i(\alpha)=\frac{\Delta_i^+(\alpha)-\Delta_i^-(\alpha)}{2}=a_i\cdot\frac{r(\alpha)-\ell(\alpha)}{2}.$$
- In this case, we have
$$[\Delta^-(\alpha),\Delta^+(\alpha)]=[\widetilde{\Delta}(\alpha)-\Delta(\alpha),\widetilde{\Delta}(\alpha)+\Delta(\alpha)],\ \text{where}$$
$$\widetilde{\Delta}(\alpha)=\sum_{i=1}^n c_i\cdot\widetilde{\Delta}_i(\alpha)=\sum_{i=1}^n c_i\cdot\left[-a_i\cdot\frac{\ell(\alpha)+r(\alpha)}{2}\right]=A\cdot\frac{\ell(\alpha)+r(\alpha)}{2}.$$
- Here, for some small h, we denoted
$$A\stackrel{\text{def}}{=}-\sum_{i=1}^n c_i\cdot a_i=\frac{f(b_1-a_1\cdot h,\ldots,b_n-a_n\cdot h)-\widetilde{y}}{h}.$$
- Similarly, $\Delta(\alpha)=\dfrac{r(\alpha)-\ell(\alpha)}{2}\cdot B$, where $B\stackrel{\text{def}}{=}\sum\limits_{i=1}^n |c_i|\cdot a_i$.

47. Linearized Case – Important Subcase: Analysis of the Problem (cont-d)

- This expression can be computed by using the Cauchy deviate method.
- Thus, we arrive at the following algorithm.

48. Linearized Case – Important Subcase: New Algorithm

- We are given:
  - a function $f(x_1, \ldots, x_n)$,
  - values $\ell(\alpha)$ and $r(\alpha)$ describing the shape of the common membership function, and
  - values a_i and b_i describing specific membership functions for each inputs x_i.
- First, we compute the values $\widetilde{y} = f(b_1, \ldots, b_n)$ and A simply by calling the algorithm f.
- We compute the expression B by using the Cauchy deviate method.
- Then, for each α, we compute the desired range $\mathbf{y}(\alpha)$ as

$$\left[\widetilde{y} + A \cdot \frac{\ell(\alpha) + r(\alpha)}{2} - B \cdot \frac{r(\alpha) - \ell(\alpha)}{2}, \widetilde{y} + A \cdot \frac{\ell(\alpha) + r(\alpha)}{2} + B \cdot \frac{r(\alpha) - \ell(\alpha)}{2}\right]$$

49. How Faster Is This Algorithm?

- In the currently used approach, we need to use interval computation technique $A \geq 9$ times.
- In the new algorithm, we only use it once.

50. This New Algorithm Can Be Extended to a More General Case

- We considered the case when the α-cuts of all membership functions are described by a linear expression with two parameters.
- It is possible to consider more general families of membership functions, in which the linear expression depends on $p \geq 3$ parameters.
- E.g., families:
  - of all possible (not necessarily symmetric) triangular functions, or
  - of all possible trapezoid functions.
- In this case, similar formulas show that we can compute the desired range by calling Cauchy method $p - 1$ times.
- For $p < 10$, this is still better than the original method.

51. Possible Extensions (cont-d)

- This idea also takes care of the case when:
  - some membership functions belong to one family (e.g., triangular)
  - and some belong to another family (e.g, Gaussian).
- In this case, we can view both linear expressions as a particular case of a general linear formula with more parameters.
- Thus, we can use the same idea as in the previous slides.

52. Conclusions

- In many practical situations, we need to propagate fuzzy uncertainty through a data processing algorithm $y = f(x_1, \ldots, x_n)$, i.e.:
  - given fuzzy information about the algorithm's inputs x_i,
  - estimate the fuzzy uncertainty of the algorithm's output y.
- State-of-the-art techniques for such propagation rely on the fact that for every α:
  - the α-cut $\mathbf{y}(\alpha)$ of the output is equal to
  - the range of the corresponding function $y = f(x_1, \ldots, x_n)$ when inputs vary in the corresponding α-cuts $x_i \in \mathbf{x}_i(\alpha)$.
- Thus, these state-of-the-art algorithms estimate such range for several (usually $A \geq 9$) different values α.
- The main limitation of the known techniques is that this time-consuming range-computation has to be repeated A times.

53. Conclusions (cont-d)

- This means that we have to call the often-time-consuming data processing algorithm $y = f(x_1, \ldots, x_n)$ at least $A \geq 9$ times.
- In this talk, we provide new algorithms that reduce the time of this data-processing-under-fuzzy-uncertainty by an order of magnitude.

54. 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), and
  - HRD-1834620 and HRD-2034030 (CAHSI Includes).
- It was also supported by the AT&T Fellowship in Information Technology.
- It was also supported by the program of the development of the Scientific-Educational Mathematical Center of Volga Federal District No. 075-02-2020-1478.