From Single to Double Use Expressions, with Applications to Parametric Interval Linear Systems: On Computational Complexity of Fuzzy and Interval Computations

Joe Lorkowski
Department of Computer Science
University of Texas at El Paso
500 W. University
El Paso, Texas 79968, USA
Email: lorkowski@ieee.org
or: lorkowski@computer.org

NAFIPS 2011

Introduction

Interval Data Processing

- Every day, we use estimated values $\widetilde{x}_1, \ldots, \widetilde{x}_n$ to get an estimated value $\widetilde{y} = f(\widetilde{x}_1, \ldots, \widetilde{x}_n)$.
- Even if an algorithm f is exact, because of uncertainty $\widetilde{x}_i \neq x_i$ produces $\widetilde{y} \neq y$.
- Often, the only knowledge of the measurement error Δx_i is the upper bound Δ_i such that $|\Delta x_i| \leq \Delta_i$
- Then, the only knowledge we have about x_i is that x_i belongs to the interval $\mathbf{x}_i = [\widetilde{x}_i - \Delta_i, \widetilde{x}_i + \Delta_i]$.

The main problem of Interval Computation

- Different values x_i from intervals $\mathbf{x}_i$ lead, in general, to different values $y = f(x_1, \ldots, x_n)$.

- To gauge the uncertainty in y, it is necessary to find the range of all possible values of y:

$$\mathbf{y} = [\underline{y}, \overline{y}] = f(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \{f(x_1, \ldots, x_n) : x_i \in \mathbf{x}_1, \ldots, \mathbf{x}_n\}.$$

- The problem of estimating the range based on given intervals $\mathbf{x}_i$ constitutes the main problem of *interval computations*

Interval Computations

- For arithmetic operations $f(x_1, x_2)$, $x_1 \in \mathbf{X}_1$, $x_2 \in \mathbf{X}_2$ there are explicit formulas called *interval arithmetic*.

- $f(x_1, x_2)$ for add, sub, mult, & div are described by:

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

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

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

$$\frac{[\underline{x}_1, \overline{x}_1]}{[\underline{x}_2, \overline{x}_2]} = [\underline{x}_1, \overline{x}_1] \cdot \frac{1}{[\underline{x}_2, \overline{x}_2]} \text{ if } 0 \notin [\underline{x}_2, \overline{x}_2];$$

$$\frac{1}{[\underline{x}_2, \overline{x}_2]} = \left[\frac{1}{\overline{x}_2}, \frac{1}{\underline{x}_2}\right] \text{ if } 0 \notin [\underline{x}_2, \overline{x}_2]$$

.

Fuzzy Data Processing

- When estimates $\widetilde{x}_i$ come from experts in the form "approximately 0.1" there are no guaranteed upper bounds on the estimation error $\Delta x_i = \widetilde{x}_i - x_i$.
- *Fuzzy Logic* is a formalization of natural language specifically designed to deal with expert estimates.
- To describe a fuzzy property $P(U)$, assign to every object $x_i \in U$, the degree $\mu_P(x_i) \in [0, 1]$ which, according to an expert, x_i satisfies the property
  - if the expert is absolutely sure it does, the degree is 1
  - if the expert is absolutely sure it does not, the degree is 0
  - else, the degree is between 0 and 1
- $\mu_P(x_i)$ can be a table lookup or a calculated value using a predefined function based on the experts' estimates.

Fuzzy Data Processing

- A real number $y = f(x_1, \ldots, x_n)$ is possible $\Leftrightarrow$

$$\exists x_1 \ldots \exists x_n((x_1 \text{ is possible}) \,\&\, \ldots \,\&\, (x_n \text{ is possible}) \,\&$$

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

- Once the degrees $\mu_i(x_i)$ (corresponding to "x_i is possible") are known, predetermined "and" and "or" operations like $f_\&(d_1, d_2) = \min(d_1, d_2)$ and $f_\vee(d_1, d_2) = \max(d_1, d_2)$ can be used to estimate the degree $\mu(y)$ to which y is possible:

$$\mu(y) = \max\{\min(\mu_1(x_1), \ldots, \mu_n(x_n) : y = f(x_1, \ldots, x_n)\}.$$

(Zadeh's extension principle)

From a computational viewpoint, fuzzy data processing can be reduced to interval data processing.

- An *alpha-cut* ($X_i(\alpha)$) is an alternative way to describe a membership function $\mu_i(x_i)$. For each $\alpha \in [0, 1]$

$$X_i(\alpha) = \{x_i : \mu_i(x_i) \geq \alpha\}$$

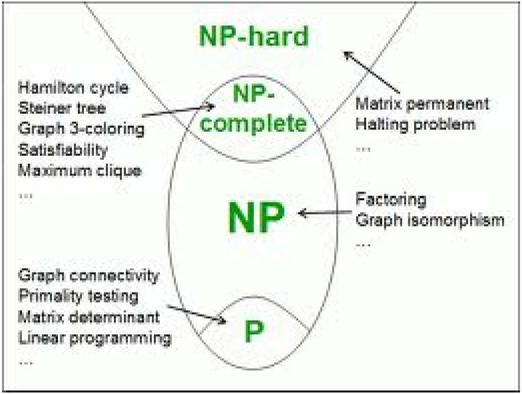
- For alpha-cuts, Zadeh's extension principle takes the following form: if $y = f(x_1, \ldots, x_n)$ then for every α, we have

$$Y(\alpha) = \{f(x_1, \ldots, x_n) : x_i \in X_i(\alpha)\}.$$

- Compare this to the main problem of interval computations

$$\mathbf{y} = [\underline{y}, \overline{y}] = \{f(x_1, \ldots, x_n) : x_1 \in \mathbf{x}_1, \ldots, \mathbf{x}_n\}.$$

What is NP Hard?



If $P \neq NP$ (as most people in CS believe), then NP-Hard problems cannot be solved in time bounded by the polynomial of the length of the input.

What is NP Hard?

- In general, the main problem of Interval Computations is NP-hard.
- This was proven by reducing the Propositional Satisfiability (SAT) problem to Interval Computations
- There are many NP-Hardness results related to Interval Computation.
- Recent work showed that some simple interval computation problems are NP-hard: e.g., the problem of computing the range of sample variance under interval uncertainty

$$V = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - E)^2, \text{ where } E = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i$$

Single Use Expressions (SUE)

- A SUE expression is one in which each variable is used at most once. Examples of SUE are:

SUE	**Not SUE**
$a \cdot (b + c)$	$a \cdot b + a \cdot c$
$\frac{1}{1+x_2/x_1}$	$\frac{x_1}{x_1+x_2}$
(For propositional formulas)	
$(v_1 \vee \neg v_2 \vee v_3) \,\&\, (\neg v_4 \vee v_5)$	$(v_1 \vee \neg v_2 \vee v_3) \,\&\, (v_1 \vee \neg v_4 \vee v_5)$

- Single Use Expressions (SUE) is a known case when naive interval computations lead to an exact range.

(ASIDE) Naive Interval Computations

- Example $y = x \cdot (1 - x)$ where $x \in [0, 1]$

- First parse the expression into elementary operations
  - $r_1 = 1 - x$
  - $y = x \cdot r_1$

- and then apply interval arithmetic to each step
  - $r_1 = [1, 1] - [0, 1] = [1, 1] + [-1, 0] = [0, 1]$
  - $y = [0, 1] \cdot [0, 1] = [\min(0, 0, 0, 1), \max(0, 0, 0, 1)] = [0, 1]$

- $[0, 1]$ is an enclosure for the exact range $[0, 0.25]$.

Naive Interval Computations works for SUE case

- Example of $y = \dfrac{x_1}{x_1 + x_2}$ converted to SUE.

  $$\frac{1}{1 + \dfrac{x_2}{x_1}} \quad \text{where } x_1 \in [1, 3], x_2 \in [2, 4]$$

- First parse the expression into elementary operations

  $r_1 = x_2 / x_1$
  $r_2 = 1 + r_1$
  $y = 1 / r_2$

- and then apply interval arithmetic to each step

  $$r_1 = \frac{[2,4]}{[1,3]} = [2,4] \cdot \frac{1}{[1,3]} = [2,4] \cdot \left[\frac{1}{3}, \frac{1}{1}\right] = [0.66, 4.0]$$
  $$r_2 = [1,1] + [0.66, 4.0] = [1.66, 5.0]$$
  $$y = \frac{1}{[1.66, 5.0]} = \left[\frac{1}{5.0}, \frac{1}{1.66}\right] = [0.2, 0.6]$$

  which is the exact range.

Double Use Expressions (DUE)

- A DUE expression is one in which each variable is used at most twice. Examples of DUE are:

DUE

$a \cdot b + a \cdot c$

$\frac{x_1}{x_1 + x_2}$

(For propositional formulas)

$(v_1 \vee \neg v_2 \vee v_3) \,\&\, (v_1 \vee \neg v_4)$

Not DUE

$a \cdot (b + c)$

$\frac{1}{1 + \frac{x_2}{x_1}}$

$(v_1 \vee \neg v_2 \vee v_3) \,\&\, (\neg v_4 \vee v_5)$

- Double Use Expressions (DUE) are known to cause excess width in naive interval computations but that does not necessarily make it NP-Hard.

Satisfiability

- Propositional Satisfiability (SAT) was the first problem proved to be NP-Hard, so it is a good tool to begin checking algorithms.
- SAT tries to make the given formula true by assigning a Boolean value to each variable.
- SAT uses propositional formulas in Conjunctive Normal Form (CNF) which are conjunctions of clauses containing disjunctions of (possibly negated) literals.
- A 3-SAT problem is a SAT problem in CNF with three variables in each clause.

$$(v_1 \vee \neg v_2 \vee v_3) \,\&\, (v_1 \vee \neg v_4 \vee v_5) \,\&\, \ldots,$$

Satisfiability of SUE

- In a SUE expression, each variable occurs only once

  $(v_1 \vee \neg v_2 \vee v_3) \,\&\, (v_4 \vee v_5 \vee \neg v_6) \,\&\, \ldots,$

- Satisfiability of SUE is easy:
  - Set one variable in every clause to evaluate to true:
  - A non-negated variable is set to true or
  - A negated variable is set to false, causing it to evaluate to true.

Satisfiability of DUE

- In a DUE expression, each variable occurs at most twice

  $(v_1 \vee \neg v_2 \vee v_3) \,\&\, (v_1 \vee v_4 \vee \neg v_5) \,\&\, \ldots,$

- Satisfiability of DUE is also done by clause elimination using an equivalent formula:
  $(v_i \vee r)\&(v_i \vee r')\&R$
  where r and r' are remainders of the clauses and R is the remainder of the expression.

- The algorithm is not much harder than SUE but longer and more tedious.

DUE in Interval Computations

- Computing the range of variance under interval uncertainty has the form
$$V = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - E)^2, \text{ where } E = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i;$$
- computing the variance with IC is NP-Hard;
- computing the variance is DUE;
- so a known NP-Hard problem reduces to DUE;
- thus, DUE under Interval Uncertainty is NP-Hard.

Interval Linear Equations

- Sometimes, there are only implicit relations between x_i and y and the simplest case is when the relations are linear and $y_1, \ldots, y_n$ are determined by

$$\sum_{j=1}^{n} a_{ij} \cdot y_j = b_i,$$

where we know interval bounds for a_{ij} and b_i

- It is known that computing the desired ranges $y_1, \ldots, y_n$ is NP-Hard when a_{ij} takes values from $\mathbf{a}_{ij}$ and b_i takes values from intervals $\mathbf{b}_i$.

- However, it is feasible to check, given values $x_1, \ldots, x_n$, if there exist values $a_{ij} \in \mathbf{a}_{ij}$ and $b_i \in \mathbf{b}_i$ for which the system is true.

- For every i, $\sum_{j=1}^{n} a_{ij} \cdot y_j$ is SUE, so its range can be found using naive interval computation.

- For every i, however, the range and the interval $\mathbf{b}_i$ must have a non-empty intersection

$$\left(\sum_{j=1}^{n} \mathbf{a}_{ij} \cdot y_j\right) \cap \mathbf{b}_i \neq \emptyset.$$

- Checking whether two intervals have an intersection is trivial:

$$[\underline{x}_1, \overline{x}_1] \cap [\underline{x}_2, \overline{x}_2] \neq \emptyset \Leftrightarrow \underline{x}_1 \leq \overline{x}_2 \,\&\, \underline{x}_2 \leq \overline{x}_1.$$

- So, there is a feasible algorithm to check if a solution satisfies the problem.

Parametric Interval Linear Systems

- Consider a *parametric* system.
  - There are k parameters $p_i, \ldots, p_k$ that take values from known intervals $\mathbf{p}_1, \ldots, \mathbf{p}_k$ and
  - values a_{ij} and b_i are linear functions of these variables

$$a_{ij} = \sum_{\ell=1}^{k} a_{ij\ell} \cdot p_\ell \text{ and } b_i = \sum_{\ell=1}^{k} b_{i\ell} \cdot p_\ell$$

- This problem is more general than the system of linear equations so finding the range for this problem is NP-Hard as well.
- However, it is possible to check whether a given tuple $x = (x_1, \ldots, x_n)$ is a solution to a given parametric interval linear system, i.e., whether there exist values p_ℓ for which $\sum_{j=1}^{n} a_{ij} \cdot y_j = b_i$.

- There is recent work by E. D. Popova showing that, if each parameter p_i occurs only in one equation (even if it occurs several times in the equation), then checking is still feasible.

- In the SUE case, consider one equation at a time – since no two equations share a parameter. For each i, the corresponding equation $\sum_{j=1}^{n} a_{ij} \cdot y_j = b_i$ takes the form

$$\sum_{j=1}^{n} \sum_{\ell=1}^{k} a_{ij\ell} \cdot y_j \cdot p_\ell = \sum_{\ell=1}^{k} b_{i\ell} \cdot p_\ell,$$

i.e., the (SUE) linear form

$$\sum_{\ell=1}^{k} A_{i\ell} \cdot p_\ell = 0, \text{ where } A_{i\ell} = \sum_{j=1}^{n} a_{ij\ell} \cdot y_j - b_{i\ell},$$

and we already know that checking the solvability of such an equation is feasible.

What If?

- What if each parameter can occur several times?
  - When only linear dependencies are allowed, there is a feasible algorithm that checks if a tuple x belongs to a solution set.
- What if each parameter can occur in only one equation but the dependence on a_{ij} and b_i on the parameters can be quadratic?
  - The problem of checking if a tuple x belongs to a solution set is NP-Hard
  - even when each parameter occurs in only one equation.

Questions?

APPENDIX A: Satisfiability of DUE Expressions

- In a DUE expression, each variable occurs at most twice

  $(v_1 \vee \neg v_2 \vee v_3) \,\&\, (v_1 \vee v_4 \vee \neg v_5) \,\&\, \ldots,$

- Satisfiability uses clause elimination similar to SUE.

- Remember, for the moment, that each clause has the form
  $(v_i \vee r)\&R$
  where r is the remainder of the clause and R is the remainder of the expression.

  - First, delete every clause containing some v_i that has a single use in the expression.

  - Second, delete pairs of clauses where v_i is either negated or non-negated in both clauses.

  - Next, delete newly-formed single use clauses.

APPENDIX A: Satisfiability of DUE Expressions

- Finally, the only remaining clauses are pairs in the form
  $(v_i \lor r)\&(\neg v_i \lor r')\&R$
  which is equivalent to a new formula
  $(r \lor r')\&R$
- If the original formula $(v_i \lor r)\&(\neg v_i \lor r')\&R$ is satisfied:
  - If v_i is true, then r' is true so $(r \lor r')$ is true.
  - If $\neg v_i$ is true, then r is true so $(r \lor r')$ is true.
- If the formula $(r \lor r')\&R$ is satisfied:
  - If r is true, then v_i is false so $(\neg v_i \lor r')$ is true.
  - If r' is true, then v_i is true so $(v_i \lor r)$ is true.
- In both cases, $(v_i \lor r)\&(\neg v_i \lor r')\&R)$ is true.
- So, DUE expressions in SAT is satisfiable.

APPENDIX B: If each parameter occurs several times

- The problem is checking whether there are values p_ℓ that satisfy the system of linear equations
  $$\sum_{\ell=1}^{k} A_{i\ell} \cdot p_\ell = 0$$
  and linear inequalities
  $$\underline{p}_\ell \le p_\ell \le \overline{p}_\ell$$
  (that describe interval constraints on p_ℓ).

- It is known that checking consistency of a given system of linear equations and inequalities is a feasible case of linear programming.

- So, any feasible algorithm for solving linear programming problems solves the above problem as well.

APPENDIX C: Dependence on Parameters is Quadratic

- What if the dependence of a_{ij} and b_i on the parameters can be quadratic

$$a_{ij} = a_{ij0} + \sum_{\ell=1}^{k} a_{ij\ell} \cdot p_\ell + \sum_{\ell=1}^{k} \sum_{\ell'=1}^{k} a_{ij\ell\ell'} \cdot p_\ell \cdot p_{\ell'};$$

$$b_i = b_{i0} + \sum_{\ell=1}^{k} b_{i\ell} \cdot p_\ell + \sum_{\ell=1}^{k} \sum_{\ell=1}^{k} b_{i\ell\ell'} \cdot p_\ell \cdot p_{\ell'}.$$

- We already know that finding the range of a quadratic function $f(p_1, \ldots, p_k)$ under interval uncertainty $p_\ell \in \mathbf{p}_\ell$, is NP-hard.

APPENDIX C: Dependence on Parameters is Quadratic

- It is also true that checking, for a given value v_0, where there exists values $p_\ell \in \mathbf{p}_\ell$ for which $f(p_1, \ldots, p_k) = v_0$ is also NP-hard.

- This NP-hard problem can be reduced to our problem by considering a very simple system consisting of a single equation:

  $a_{11} \cdot y_1 = b_1$, with $y_1 = 1$, $b_1 = v_0$, and $a_{11} = f(p_1, \ldots, p_k)$.

  The tuple $x = (1)$ belongs to the solution set if and only if there exist values p_ℓ for which $f(p_1, \ldots, p_k) = v_0$.

- So, allowing the dependence of parameters to be quadratic is NP-hard.

Bibliography

C. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, Boston, Massachusetts, 2009.

S. Ferson, L. Ginzburg, V. Kreinovich, L. Longpré, and M. Aviles, “Computing Variance for Interval Data is NP-Hard", *ACM SIGACT News*, 2002, Vol. 33, No. 2, pp. 108–118.

S. Ferson, L. Ginzburg, V. Kreinovich, L. Longpré, and M. Aviles, “Exact Bounds on Finite Populations of Interval Data", *Reliable Computing*, 2005, Vol. 11, No. 3, pp. 207–233.

A. A. Gaganov, *Computational complexity of the range of the polynomial in several variables*, Leningrad University, Math. Department, M.S. Thesis, 1981 (in Russian).

A. A. Gaganov, “Computational complexity of the range of the polynomial in several variables”, *Cybernetics*, 1985, pp. 418–421.

- E. Hansen, “Sharpness in interval computations", *Reliable Computing*, 1997, Vol. 3, pp. 7–29.
- L. Jaulin, M. Kieffer, O. Didrit, and E. Walter, *Applied Interval Analysis, with Examples in Parameter and State Estimation, Robust Control and Robotics*, Springer-Verlag, London, 2001.
- G. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*, Upper Saddle River, New Jersey: Prentice Hall, 1995.
- V. Kreinovich, A. Lakeyev, J. Rohn, and P. Kahl, *Computational complexity and feasibility of data processing and interval computations*, Kluwer, Dordrecht, 1998.
- R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to Interval Analysis*, SIAM Press, Philadelphia, Pennsylviania, 2009.
- H. T. Nguyen and E. A. Walker, *First Course on Fuzzy Logic*, CRC Press, Boca Raton, Florida, 2006.

E. D. Popova, "Explicit Characterization of a Class of Parametric Solution Sets", *Compt. Rend. Acad. Bulg. Sci.*, 2009, Vol. 62, No. 10, pp. 1207–1216.

E. D. Popova, "Characterization of paremetric solution sets", *Abstracts of the 14th GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic, and Validated Numerics SCAN'2010*, Lyon, France, September 27–30, 2010, pp. 118–119.

S. Rabinovich, *Measurement Errors and Uncertainties: Theory and Practice*, Springer Verlag, New York, 2005.