An efficient midpoint-radius implementation to handle symmetric fuzzy intervals

Manuel Marin, David Defour and Federico Milano

DALI, University of Perpignan Via Domitia,
Electricity Research Centre, University College Dublin
**Introduction**

- Interval arithmetic is a useful tool to provide reliable results in computations with uncertain data (e.g., the speed is between 100 and 120 km/h).

- In addition, *fuzzy* interval arithmetic provides an answer when the information in the knowledge base is more ambiguous and imprecise (e.g., the speed is high).

- Most fuzzy interval implementations are based on the lower-upper representation format. *This talk discusses the use of midpoint-radius to improve performance when dealing with symmetric fuzzy intervals.*
1 Mathematical background
   - Fuzzy interval arithmetic
   - Representation formats

2 Implementation

3 Tests and results
**Fuzzy Intervals**

Fuzzy intervals are characterized by a membership function, $\mu : \mathbb{R} \to [0, 1]$. For each $\alpha \in [0, 1]$, an $\alpha$-cut is obtained:
Fuzzy intervals are characterized by a membership function, $\mu : \mathbb{R} \to [0, 1]$. For each $\alpha \in [0, 1]$, an $\alpha$-cut is obtained:

If the fuzzy interval is symmetric, then all the $\alpha$-cuts have the same midpoint.
Fuzzy arithmetic operations are decomposed into a series of interval operations, one per $\alpha$-cut:
Fuzzy arithmetic operations are decomposed into a series of interval operations, one per $\alpha$-cut:

\[\begin{array}{c}
\begin{array}{c}
\tilde{x} \\
\text{addition} \\
\text{subtraction} \\
\text{multiplication}
\end{array}
\end{array}\]

Addition, subtraction and multiplication of fuzzy intervals preserve symmetry.
LOWER-UPPER REPRESENTATION

\[ [x, \bar{x}] = \{x \in \mathbb{R}: x \leq x \leq \bar{x}\}. \]
Lower-upper representation

\[ [x, \overline{x}] = \{x \in \mathbb{R}: x \leq x \leq \overline{x} \}. \]

Rounded interval arithmetic in the lower-upper format

\[
x + y = [\nabla (x + y), \triangle (\overline{x} + \overline{y})],
\]
\[
x - y = [\nabla (x - \overline{y}), \triangle (\overline{x} - y)],
\]
\[
x \cdot y = [\nabla (\min (xy, xy, \overline{xy}, \overline{xy})), \triangle (\max (xy, xy, \overline{xy}, \overline{xy}))],
\]
\[
\frac{1}{y} = [\nabla \left( \frac{1}{y} \right), \triangle \left( \frac{1}{\overline{y}} \right)], \quad 0 \notin y.
\]

\(\nabla\): rounding downwards
\(\triangle\): rounding upwards
Midpoint-radius representation

\[ \langle \tilde{x}, \rho_x \rangle = \{ x \in \mathbb{R} : |x - \tilde{x}| \leq \rho_x \} \]
Midpoint-radius representation

\[ \langle \tilde{x}, \rho_x \rangle = \{ x \in \mathbb{R} : |x - \tilde{x}| \leq \rho_x \} \]

Rounded interval arithmetic in the midpoint-radius format

\[
\begin{align*}
x \pm y &= \langle \square(\tilde{x} \pm \tilde{y}), \triangle(\epsilon' |\tilde{x} \pm \tilde{y}| + \rho_x + \rho_y) \rangle, \\
x \cdot y &= \langle \square(\tilde{x}\tilde{y}), \triangle(\eta + \epsilon' |\tilde{x}\tilde{y}| + (|\tilde{x}| + \rho_x)\rho_y + |\tilde{y}|\rho_x) \rangle, \\
\frac{1}{y} &= \langle \square\left(\frac{1}{\tilde{y}}\right), \triangle\left(\eta + \epsilon' \left|\frac{1}{\tilde{y}}\right| + \frac{-\rho_y}{|\tilde{y}|(\rho_y - |\tilde{y}|)}\right) \rangle, \quad 0 \notin y.
\end{align*}
\]

\(\square\): rounding to nearest
\(\triangle\): rounding upwards
\(\epsilon'\): relative rounding error divided by 2
\(\eta\): smallest representable floating-point number
Outline

1. Mathematical background
   - Fuzzy interval arithmetic
   - Representation formats

2. Implementation

3. Tests and results
Algorithm 1 Fuzzy multiplication in the lower-upper format.

**Input:** Fuzzy operands $x$ and $y$.
**Output:** Fuzzy result $z = x \cdot y$.

1: for $i$ in $1, \ldots, N$ do 
2: $z_i = \nabla (\min(xy, xy, xy, xy))$ 
3: $\bar{z}_i = \Delta (\max(xy, xy, xy, xy))$

Requires $14N$ floating-point operations.
**Algorithms**

**Algorithm 1** Fuzzy multiplication in the lower-upper format.

**Input:** Fuzzy operands $x$ and $y$.  
**Output:** Fuzzy result $z = x \cdot y$.  
1: for $i$ in $1, \ldots, N$ do  
2: $z_i = \nabla(\min(xy, x\bar{y}, \bar{x}y, \bar{x}\bar{y}))$  
3: $\bar{z}_i = \Delta(\max(xy, x\bar{y}, \bar{x}y, \bar{x}\bar{y}))$

Requires $14N$ floating-point operations.

**Algorithm 2** Symmetric fuzzy multiplication in the midpoint-radius format.

**Input:** Symmetric fuzzy operands $x$ and $y$.  
**Output:** Symmetric fuzzy result $z = x \cdot y$.  
1: $\tilde{z} = \Box(\tilde{x}\tilde{y})$  
2: for $i$ in $1, \ldots, N$ do  
3: $\rho_{z,i} = \Delta(\eta + c'|\tilde{z}| + (|\tilde{x}| + \rho_{x,i})\rho_{y,i} + |\tilde{y}|\rho_{x,i})$

Requires $6 + 5N$ floating-point operations.
### Performance Considerations

**Table 1:** Number of floating-point instructions per arithmetical operation, for different data types.

<table>
<thead>
<tr>
<th>Data type</th>
<th>Addition</th>
<th>Multiplication</th>
<th>Inversion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalar</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Lower-upper interval</td>
<td>2</td>
<td>14</td>
<td>2</td>
</tr>
<tr>
<td>Midpoint-radius interval</td>
<td>5</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>Lower-upper fuzzy</td>
<td>$2N$</td>
<td>$14N$</td>
<td>$2N$</td>
</tr>
<tr>
<td>Midpoint-radius fuzzy</td>
<td>$3 + 2N$</td>
<td>$6 + 5N$</td>
<td>$5 + 5N$</td>
</tr>
</tbody>
</table>

**Table 2:** Memory requirements of different data types.

<table>
<thead>
<tr>
<th>Data type</th>
<th>Memory usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalar</td>
<td>1</td>
</tr>
<tr>
<td>Lower-upper interval</td>
<td>2</td>
</tr>
<tr>
<td>Midpoint-radius interval</td>
<td>2</td>
</tr>
<tr>
<td>Lower-upper fuzzy</td>
<td>$2N$</td>
</tr>
<tr>
<td>Midpoint-radius fuzzy</td>
<td>$1 + N$</td>
</tr>
</tbody>
</table>

$N$: number of $\alpha$-cuts
Fuzzy intervals can also be represented in terms of midpoint and radius increments:

$$\rho_i = \sum_{k=1}^{i} \delta_k = \rho_{i-1} + \delta_i.$$
### Algorithms

**Algorithm 3** Symmetric fuzzy addition in the midpoint-increment format.

**Input:** Symmetric fuzzy operands $x$ and $y$.

**Output:** Symmetric fuzzy result $z = x + y$.

1. $\tilde{z} = \Box(\tilde{x} + \tilde{y})$
2. $\delta_{z,1} = \triangle\left(\frac{1}{2}\epsilon|\tilde{z}| + \delta_{x,1} + \delta_{y,1}\right)$
3. **for** $i$ in 2, $\ldots$, $N$ **do**
   4. $\delta_{z,i} = \triangle(\delta_{x,i} + \delta_{y,i})$

Requires $1 + 4N$ operations, fewer than midpoint-radius.
### Algorithm 4 Symmetric fuzzy multiplication in the midpoint-increment format.

**Input:** Symmetric fuzzy operands $x$ and $y$.  
**Output:** Symmetric fuzzy result $z = x \cdot y$.  

1. $z = \Box(\tilde{x}\tilde{y})$
2. $t_{x,1} = \triangle(|\tilde{x}| + \delta_{x,1})$
3. $t_{y,1} = \triangle(|\tilde{y}|)$
4. $\delta_{z,1} = \triangle(\eta + \frac{1}{2}\epsilon|\tilde{z}| + t_{x,1}\delta_{y,1} + t_{y,1}\delta_{x,1})$
5. for $i$ in $2, \ldots, N$ do
   6. $t_{x,i} = \triangle(t_{x,i-1} + \delta_{x,i})$
   7. $t_{y,i} = \triangle(t_{y,i-1} + \delta_{y,i})$
   8. $\delta_{z,i} = \triangle(\eta + \frac{1}{2}\epsilon|\tilde{z}| + t_{x,i}\delta_{y,i} + t_{y,i}\delta_{x,i})$

Requires $6 + 5N$ operations, the same as midpoint-radius.
Performance considerations

Table 3: Number of floating-point instructions per arithmetical operation, for different data types.

<table>
<thead>
<tr>
<th>Data type</th>
<th>Addition</th>
<th>Multiplication</th>
<th>Inversion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalar</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Lower-upper interval</td>
<td>2</td>
<td>14</td>
<td>2</td>
</tr>
<tr>
<td>Midpoint-radius interval</td>
<td>5</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>Lower-upper fuzzy</td>
<td>$2N$</td>
<td>$14N$</td>
<td>$2N$</td>
</tr>
<tr>
<td>Midpoint-radius fuzzy</td>
<td>$3 + 2N$</td>
<td>$6 + 5N$</td>
<td>$5 + 5N$</td>
</tr>
<tr>
<td>Midpoint-increment fuzzy</td>
<td>$4 + N$</td>
<td>$6 + 5N$</td>
<td>$5 + 5N$</td>
</tr>
</tbody>
</table>

Table 4: Memory requirements of different data types.

<table>
<thead>
<tr>
<th>Data type</th>
<th>Memory usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalar</td>
<td>1</td>
</tr>
<tr>
<td>Lower-upper interval</td>
<td>2</td>
</tr>
<tr>
<td>Midpoint-radius interval</td>
<td>2</td>
</tr>
<tr>
<td>Lower-upper fuzzy</td>
<td>$2N$</td>
</tr>
<tr>
<td>Midpoint-radius fuzzy</td>
<td>$1 + N$</td>
</tr>
<tr>
<td>Midpoint-increment fuzzy</td>
<td>$1 + N$</td>
</tr>
</tbody>
</table>

$N$: number of $\alpha$-cuts
**ERROR ANALYSIS**

**Absolute error in the midpoint-radius format ($i \geq 2$)**

$$\rho_{z,i} = (\epsilon' |\tilde{z}| + \rho_{x,i}) + \rho_{y,i}. $$

$$E(\rho_{z,i}) \leq \epsilon (|\epsilon' |\tilde{z}| + \rho_{x,i}) + |\epsilon' |\tilde{z}| + \rho_{x,i} + \rho_{y,i}|$$

$$= \epsilon (\epsilon |\tilde{z}| + 2\rho_{x,i} + \rho_{y,i}).$$
**ERROR ANALYSIS**

**Absolute error in the midpoint-radius format \((i \geq 2)\)**

\[
\rho_{z,i} = (\epsilon'|\ddot{z}| + \rho_{x,i}) + \rho_{y,i}.
\]

\[
E(\rho_{z,i}) \leq \epsilon \left( |\epsilon'||\dddot{z}| + \rho_{x,i} \right) + |\epsilon'||\ddot{z}| + \rho_{x,i} + \rho_{y,i} |)
= \epsilon \left( |\dddot{z}| + 2\rho_{x,i} + \rho_{y,i} \right).
\]

**Absolute error in the midpoint-increment format \((i \geq 2)\)**

\[
\rho_{z,i} = \rho_{z,i-1} + \delta_{z,i}.
\]

\[
E(\rho_{z,i}) = E(\rho_{z,i-1}) + E(\delta_{z,i})
\leq \epsilon \left( |\dddot{z}| + 2\rho_{x,i-1} + \rho_{y,i-1} \right) + \epsilon |\delta_{x,i} + \delta_{y,i}|
= \epsilon \left( |\dddot{z}| + \rho_{x,i-1} + \rho_{x,i} + \rho_{y,i} \right).
\]
Outline

1 Mathematical background
   • Fuzzy interval arithmetic
   • Representation formats

2 Implementation

3 Tests and results
**Compute-bound benchmark: AXPY loop**

![Graph (a)](image1.png)

![Graph (b)](image2.png)

**Figure 1:** (a) Performance comparison of different representation formats and architectures. (b) Speed-up of midpoint-radius over lower-upper.

For fewer than 8 $\alpha$-cuts the speed-up curve follows the theoretical ratio:

$$\frac{16N}{9 + 7N}$$
Tests and results

Memory-bound benchmark: RADIX sort

(a) Performance comparison of different representation formats and architectures.
(b) Speed-up of midpoint-radius over lower-upper.

Figure 2: (a) Performance comparison of different representation formats and architectures. (b) Speed-up of midpoint-radius over lower-upper.

For fewer than 7 $\alpha$-cuts the speed-up curve follows the theoretical ratio:

\[
\frac{2N}{1 + N}
\]
CONCLUSIONS AND FUTURE WORK

- Midpoint-radius representation is an attractive alternative to handle symmetric fuzzy intervals. It achieves a speed-up of 2 to 20 over lower-upper in both compute and memory-bound benchmarks.


- Future work will consider the implementation of the midpoint-increment format and further assessment of accuracy issues.