Efficient Ranking and Unranking of Discrete Fuzzy Numbers

Manuel González-Hidalgo[1,2], Arnau Mir[1], Alejandro Mus[1], and Juan Vicente Riera[1]

[1] Soft Computing, Image Processing and Aggregation Research Group (SCOPIA), Artificial Intelligence Research, Institute of the Balearic Islands (IAIB), Health Research Institute of the Balearic Islands (IdISBa), Mathematics and Computer Science Department, University of the Balearic Islands, Spain
{manuel.gonzalez,arnau.mir,alejandro.mus,jvicente.riera}@uib.es
[2] Laboratory of Artificial Intelligence Applications (LAIA@UIB)

Abstract. Discrete fuzzy numbers are a useful representation of linguistic information, but recent constructions based on total orders require repeated evaluations of the position map and its inverse. A direct implementation enumerates or sorts the whole ordered set, whose size grows combinatorially with the number of membership levels. This ongoing research develops exact ranking and unranking algorithms for discrete fuzzy numbers on a finite chain. The approach exploits nested α-cuts and closed-form combinatorial counts while avoiding exhaustive enumeration. The resulting procedures compute pos and pos^{-1} in $O(n^2 m \log n)$ time with constant extra memory. Preliminary experiments confirm the expected linear scaling in the number of membership values for fixed n.

Keywords: Discrete fuzzy numbers · Total orders · Ranking · Unranking · Computational efficiency

1 Motivation

Discrete fuzzy numbers (DFNs) are widely used in computing-with-words and decision-making models because they combine finite linguistic chains with gradual membership information [1–3].

Let $\mathcal{D}_1^{L_n \to Y_m}$ denote the finite family of dfns with support in a closed subinterval of $L_n = \{0, 1, \ldots, n\}$ and membership values in $Y_m = \{0 = y_1 < y_2 < \cdots < y_m = 1\}$. Its cardinality is [1]

$$|\mathcal{D}_1^{L_n \to Y_m}| = \binom{n + 2m - 2}{2m - 2}. \tag{1}$$

A total order on intervals can be extended to a total order $\preceq_\delta^{\Delta\downarrow}$ on $\mathcal{D}_1^{L_n \to Y_m}$, yielding a finite chain of cardinality $|\mathcal{D}_1^{L_n \to Y_m}|$. This enables the definition of

aggregation and implication functions on DFNs by transporting operators from $L_k = \{0, \ldots, k\}$, where $k = |\mathcal{D}_1^{L_n \to Y_m}| - 1$ [4, 5, 1], through the *position map*

$$\mathrm{pos} : \mathcal{D}_1^{L_n \to Y_m} \to L_k, \qquad \mathrm{pos}(A) = \left|\{X \in \mathcal{D}_1^{L_n \to Y_m} \mid X \preceq_\delta^{\Delta\downarrow} A\}\right| - 1,$$

an order isomorphism between $(\mathcal{D}_1^{L_n \to Y_m}, \preceq_\delta^{\Delta\downarrow})$ and $(L_k, \leq)$ [1].

However, the computation of $pos(A)$ suggests a brute-force approach requiring the generation, sorting, and search of all DFNs, which quickly becomes infeasible as m grows. The aim of this work is to compute the ranking $\mathrm{pos}(A)$ and unranking $\mathrm{pos}^{-1}(i)$ exactly, without materialising the complete ordered space.

2 Explicit pseudocode for ranking and unranking

Let $\mathcal{I}(L_n)$ be the set of closed intervals $[a, b] \subseteq L_n$, ordered by a fixed total order $\preceq$ defined on this set. Since $|\mathcal{I}(L_n)| = (n+1)(n+2)/2$, every α-cut of $A \in \mathcal{D}_1^{L_n \to Y_m}$ can be treated as an element of a finite interval chain. The α-cut notation and its reconstruction role are detailed in [1]. The following algorithms consider all levels of each DFN, beginning at $y_m = 1$ and going down to $y_1 = 0$, using the nesting property of α-cuts.

For an interval $[a, b]$ occupying the j-th position in $\mathcal{I}(L_n)$, let $\mathrm{SDFN}(a, b, j)$ denote the set of generalized DFNs whose maximum membership value is y_j exactly on $[a, b]$, while the remaining values belong to $\{y_1, \ldots, y_j\}$. The consideration of these sets is a key idea in our algorithms. Their cardinality [1] is given by

$$|\,\mathrm{SDFN}(a, b, j)| = \binom{a+j-2}{j-2}\binom{n-b+j-2}{j-2} \tag{2}$$

The formula separates the choices to the left and right of the core interval: the left branch is nondecreasing and the right branch is nonincreasing. Hence, the number of DFNs below a candidate α-cut can be computed by summing products of binomial coefficients instead of enumerating objects.

Based on the previous considerations, the two procedures below are written explicitly as pseudocode blocks, following the ranking–unranking construction of [1]. Both use the same ordered block decomposition of $\mathcal{D}_1^{L_n \to Y_m}$. Unranking locates the residual index within the first compatible block whose cumulative size reaches its one-based residual rank, while ranking accumulates the sizes of all compatible blocks preceding the observed α-cut.

Algorithm 1. Unranking algorithm for $\mathrm{pos}^{-1}(i)$**.**
Input: index $i \in \{0, \ldots, |\mathcal{D}_1^{L_n \to Y_m}| - 1\}$; ordered intervals $I_1 \prec \cdots \prec I_{|\mathcal{I}(L_n)|}$.
Output: the DFN A such that $\mathrm{pos}(A) = i$.

$r \leftarrow i + 1$; $A^{y_{m+1}} \leftarrow \emptyset$; $k_{m+1} \leftarrow |\mathcal{I}(L_n)|$.
for $j = m, m-1, \ldots, 1$ **do**
  $C_j \leftarrow \{I_\ell \mid A^{y_{j+1}} \subseteq I_\ell,\ \ell \leq k_{j+1}\}$.
  write $C_j = \{I_1^{(j)} \prec \cdots \prec I_{s_j}^{(j)}\}$, with $I_h^{(j)} = [a_h, b_h]$.
  find the least q such that $\sum_{h=1}^{q} |\,\mathrm{SDFN}(a_h, b_h, j)| \geq r$.
  $A^{y_j} \leftarrow I_q^{(j)}$; $k_j \leftarrow q$.
  $r \leftarrow r - \sum_{h<q} |\,\mathrm{SDFN}(a_h, b_h, j)|$.
end for
return reconstruct$(A^{y_1}, \ldots, A^{y_m})$.

Algorithm 2. Ranking algorithm for $\mathrm{pos}(A)$.
Input: a DFN $A \in \mathcal{D}_1^{L_n \to Y_m}$; ordered intervals $I_1 \prec \cdots \prec I_{|\mathcal{I}(L_n)|}$.
Output: the index $i = \mathrm{pos}(A)$.

$r \leftarrow 0$; compute the α-cuts A^{y_j}, $j = 1, \ldots, m$.
$A^{y_{m+1}} \leftarrow \emptyset$.
for $j = m, m-1, \ldots, 1$ **do**
  $C_j \leftarrow \{I_\ell \mid A^{y_{j+1}} \subseteq I_\ell,\ I_\ell \preceq A^{y_j}\}$.
  for each $I_\ell = [a_\ell, b_\ell] \in C_j \setminus \{A^{y_j}\}$ **do**
    $r \leftarrow r + |\,\mathrm{SDFN}(a_\ell, b_\ell, j)|$.
  end for
end for
return r.

The block sizes used in both algorithms are exactly the values in Eq. (2). In Algorithm 2 the set C_j includes the observed cut A^{y_j}, but the loop sums only the blocks that precede it. The procedures are mutually consistent because they traverse the same nested blocks in opposite directions. At each membership level, the candidate interval set has size at most $O(n^2)$. If it is sorted or searched according to the chosen interval order, the cost per level is $O(n^2 \log n)$. Thus, both algorithms run in

$$O(n^2 m \log n), \tag{3}$$

using constant extra memory apart from the representation of the input or output DFN. This replaces brute-force enumeration over $|\mathcal{D}_1^{L_n \to Y_m}|$ elements, whose dependence on m is combinatorial.

3 Preliminary empirical evidence

The full execution-time table is more informative than a single example, because it shows the practical scaling of both routines across one order of magnitude in the number of membership levels. We evaluated the algorithms in Python using the interval order t-inc. The experiments fixed $n = 10$, varied m from 100 to 1000, and used $K = 500$ uniformly sampled indices per configuration. Correctness was checked by verifying $\mathrm{pos}(\mathrm{pos}^{-1}(i)) = i$ in every trial. Table 1 reports the mean runtime and standard deviation for unranking and ranking, following [1].

Table 1. Average execution time as a function of m for fixed $n = 10$. Results over $K = 500$ trials; order=t-inc.

m	unrank $\bar{t}_m$ (ms)	unrank s_m (ms)	rank $\bar{t}_m$ (ms)	rank s_m (ms)
100	0.870	0.172	0.810	0.173
200	1.713	0.322	1.565	0.310
300	2.660	0.503	2.426	0.481
400	3.554	0.768	3.255	0.742
500	4.486	0.932	4.113	0.888
600	5.392	1.080	4.955	1.049
700	6.253	1.300	5.746	1.263
800	7.207	1.396	6.606	1.349
900	7.924	1.600	7.272	1.553
1000	8.960	1.822	8.205	1.739

The table confirms the expected near-linear growth in m for fixed n, in agreement with the m-dependence predicted by Eq. 3 for fixed n. A linear fit for $m \geq 400$ gives slopes 8.90×10^{-3} and 8.15×10^{-3} ms per level for unranking and ranking, with intercepts about 3×10^{-2} ms. Thus, for $n = 10$, the

measured per-level cost is about 8–9 microseconds. This evidence supports using the routines inside position-based operators of the form $F_{\mathcal{D}_1^{L_n \to Y_m}}(A, B) = \mathrm{pos}^{-1}(F(\mathrm{pos}(A), \mathrm{pos}(B)))$, where repeated rank/unrank calls replace a precomputed table of the whole lattice.

4 Conclusions and ongoing work

This work turns the position isomorphism used in recent algebraic constructions on DFNs into an operational indexing mechanism. The proposed procedures compute pos and pos^{-1} exactly, avoid storing the whole set $\mathcal{D}_1^{L_n \to Y_m}$, and reduce the dependence on the number of membership levels to a linear one for fixed n. This is relevant when position-based implication and aggregation functions must be evaluated many times. Ongoing work focuses on optimising the implementation for several interval orders, integrating the routines into position-based logical operators, and extending the same strategy to discrete Z-numbers, where value and reliability components introduce a richer two-level ordering problem. A complementary benchmark will compare against explicit enumeration, including peak memory, to quantify the practical break-even point.

Acknowledgments

The authors would like to acknowledge the support of the *Programa de Foment de la Recerca i la Innovació de la Universitat de les Illes Balears 2024–2026.*

References

1. Mir, A., Mus, A., Riera, J.V.: An Efficient Computational Framework for Discrete Fuzzy Numbers Based on Total Orders. arXiv:2511.17080 (2025). doi:10.48550/arXiv.2511.17080
2. Voxman, W.: Canonical representations of discrete fuzzy numbers. Fuzzy Sets and Systems **118**, 457–466 (2001)
3. Massanet, S., Riera, J.V., Torrens, J., Herrera-Viedma, E.: A new linguistic computational model based on discrete fuzzy numbers for computing with words. Information Sciences **258**, 277–290 (2014)
4. Riera, J.V., Massanet, S., Bustince, H., Fernandez, J.: On admissible orders on the set of discrete fuzzy numbers for application in decision making problems. Mathematics **9**, 95 (2021)
5. González-Hidalgo, M., Massanet, S., Mir, A., Riera, J.V., Miguel, L.D.: On fuzzy implication functions based on admissible orders on the set of discrete fuzzy numbers. International Journal of Computational Intelligence Systems **18**, 130 (2025)
6. Munar, M., Massanet, S., Ruiz-Aguilera, D.: On the cardinality of some families of discrete connectives. Information Sciences **621**, 708–728 (2023)