Optimizing Cardinality-aware Combination Rules in Belief Functions Theory: an Enhanced Framework

Faouzi Sebbak
Ecole Militaire Polytechnique
BP 17, Bordj El Bahri, 16046,
Algiers, Algeria
Email: faouzi.sebbak@gmail.com

Mustapha Read Senouci
Ecole Militaire Polytechnique
BP 17, Bordj El Bahri, 16046,
Algiers, Algeria
Email:mrsenouci@gmail.com

***Abstract*—Belief Functions Theory (BFT), also known as Dempster-Shafer theory, has emerged as a powerful framework for uncertain modeling and reasoning in various domains. At the heart of BFT, lies the concept of combination rules, which govern the fusion process and have a profound impact on the quality and reliability of the results. Currently, there is a noticeable trend in the field, with a growing emphasis on cardinality-aware combination rules, reflecting the need for more nuanced and context-aware approaches to uncertainty management. Despite this, there exists a notable gap in the implementation of cardinality-aware rules within existing frameworks, limiting their applicability in complex decision-making scenarios. This paper addresses this gap by proposing an enhanced MATLAB framework that optimizes computational complexity, thereby enabling efficient implementation of both current and future cardinality-aware combination rules. By providing detailed insights into the framework's structure, key functions, and examples of the implementation of these cardinality-aware rules, this paper aims to bridge the identified gap and enhance the usability of BFT in practical applications. The source code for this enhanced framework has been shared with the community.**

***Index Terms*—Belief Functions, Cardinality-aware Combination Rules, DST MATLAB Framework.**

I. Introduction

In a world characterized by increasing complexity, decision-makers often grapple with substantial volumes of uncertain and conflicting information. Belief Functions Theory (BFT), grounded in the seminal work of Dempster and Shafer [1], [2], provides a structured framework for modeling and managing uncertainty. This theory has found applications across diverse domains, including networking [3], transportation systems [4], and smart environments [5].

At the core of BFT lies the concept of combination rules, which provide a systematic means of aggregating evidence from multiple sources to derive coherent and reliable conclusions. These rules govern the fusion process and have a profound impact on the quality and reliability of the results [6].

Recently, there has been an emphasis on cardinality-aware combination rules [7]–[10], which stand out for their mathematical integration of the cardinality operation within their computation process. This shift represents a move towards more sophisticated approaches that acknowledge the varying degrees of support from evidence and how they affect decision outcomes. Understanding and leveraging these cardinality-aware combination rules are crucial for harnessing the full potential of BFT in addressing real-world decision-making challenges across a wide range of domains. However, current implementations [11]–[14], especially the widely adopted framework from [14] endorsed by the belief functions society[1], face challenges in efficiently handling the integration of cardinality-aware rules. Indeed, the complexity associated with incorporating these rules into existing implementations presents a significant hurdle, underlining a crucial gap that must be addressed to improve the practicality and applicability of BFT in decision-making scenarios.

To address the aforementioned gap in current implementations, this paper proposes enhancing the widely-utilized framework [14] by introducing new functions aimed at optimizing its computational complexity, instead of creating an entirely new framework. This enhancement facilitates the seamless and optimized implementation of both existing and new cardinality-aware combination rules, thereby providing a more efficient and versatile framework for leveraging these rules within BFT. We note that Dempster's rule is not considered by this enhancement. The source code for this enhanced framework has been shared with the community[2].

The remainder of this paper is organized as follows. Section II covers some background information related to the belief functions theory and basic combination rules. In Section III, a taxonomy of recent combination rules is presented to provide a comprehensive understanding of the current landscape. Section IV discusses the proposed enhanced framework, while Section V focuses on evaluating its performance. Finally, Section VI concludes the paper.

II. Background on BFT

This section covers some background information related to the belief functions theory and basic combination rules.

A. Representation of Information

1) Frame of Discernment: BFT represents a problem domain by a set called the Frame of Discernment (FoD) Θ, which

[1] https://www.bfasociety.org/
[2] https://github.com/LeSinus/EDST

is the set of possible states of the system.

$$\Theta = \{\theta_1 \dots \theta_k \dots \theta_N\} = \bigcup_{k=1}^{N} \{\theta_k\}$$

The elements of the power set[3] 2^Θ are called *hypotheses*.

2) Basic Belief Assignment: The central element of BFT is the basic belief assignment (BBA) or mass function. A BBA is a function m from the power set 2^Θ to $[0, 1]$ satisfying[4]:

$$\begin{aligned} m(\varnothing) &= 0 \\ \sum_{A:A\subseteq\Theta} m(A) &= 1 \end{aligned} \tag{1}$$

A subset A of Θ that has a positive mass is referred to as a *focal set element* of $m(\cdot)$. The collection of all focal elements is known as the *core function*.

3) Belief and Plausibility Functions: Two other evidential functions can be used to represent an agent's belief. The belief function (Bel) represents the belief assigned to an event $A \subseteq \Theta$ given the available evidence. It is obtained by summing all the basic belief masses $m(B)$ for $B \subseteq A, B \neq \varnothing$. We have:

$$Bel(A) = \sum_{B:\varnothing\neq B\subseteq A} m(B),\ \forall A \subseteq \Theta, A \neq \varnothing \tag{2}$$

where $Bel(\varnothing) = 0$, and $Bel(\Theta) = 1$. The belief function is also called the lower probability function. The dual of Bel, called plausibility function (Pl) or upper probability function is defined as:

$$Pl(A) = \sum_{X\subseteq\Theta:X\cap A\neq\varnothing} m(X),\ \forall A \subseteq \Theta \tag{3}$$

The quantity Pl is thus equal to the sum of the basic belief masses assigned to propositions that are not in contradiction with A. It corresponds to the maximum degree of support that could be given to A.

Other functions have been defined such as the *commonality function* Q which is generally used as a technical device to simplify proofs of computational theorems. Shafer [1] showed that a one-to-one correspondence exists between mass, belief, plausibility, and commonality, meaning that given any one of m, Bel, Pl, or Q, the other three can be calculated. Thus, a body of evidence in any of the four forms may be called a belief function since Bel is uniquely determined and can be recovered.

B. Handling Information

In addition to the different ways to represent bodies of evidence provided by the evidence theory, a set of methods and tools to handle these bodies of evidence is also provided.

1) Combination: When several pieces of evidence are obtained through distinct sources over the same FoD Θ, new evidence representing the consensus of those disparate opinions can be obtained through the combination operation. Many combination rules have been proposed such as Yager's rule [15], Zhang's center combination rule [16], and Dubois and Prade's disjunctive pooling rule [17]. In the well-known and widely used Dempster's rule of combination, two mass functions m_1 and m_2 can be combined to form $m_{DS} = m_1 \oplus m_2$ using the following formulas:

$$\begin{aligned} m_{DS}(C) &= \frac{m_{12}(C)}{1 - m_{12}(\varnothing)} \\ m_{12}(C) &= \sum_{\substack{A,B\in 2^\Theta \\ A\cap B=C}} m_1(A)\, m_2(B) \end{aligned} \tag{4}$$

2) Discounting of Information: Occasionally, sources can be susceptible to misreading or malfunctioning based on their type or deployment environment [18]. In such cases, these sources are regarded as only partially reliable. This level of source reliability can be incorporated by discounting their initial BBA using a discount rate $\gamma \in [0, 1]$. The lower the reliability is, the larger the discount rate will be.

Suppose that the BBA m on Θ represents the source belief about the actual value of Θ. The discounted m is a new BBA denoted by m^γ, defined as:

$$\begin{aligned} m^\gamma(A) &= \ (1-\gamma)m(A) \quad for \quad A \subset \Theta \\ m^\gamma(\Theta) &= \ \gamma + (1-\gamma)m(\Theta) \end{aligned} \tag{5}$$

C. Decision Making

In BFT, the usual approach to problem-solving involves collecting bodies of evidence and expressing them as belief functions. These belief functions are then merged using Dempster's rule (or other similar rules). Afterward, the best-supported hypothesis is chosen based on measures like Bel and Pl offered by the Dempster-Shafer theory. Another measure available in the transferable belief model is the Pignistic transformation [19] defined by:

$$BetP(A) = \sum_{B\subseteq\Theta} \frac{|A\cap B|}{|B|} \frac{m(B)}{1 - m(\varnothing)},\ \forall A \subseteq \Theta \tag{6}$$

where $|A|$ designs the cardinality of the set A.

III. Combination Rules

We begin this section by categorizing and establishing a taxonomy for existing rules, providing a structured understanding of their diversity and applications.

A. Taxonomy and theoretical insights

In contrast to Smets's property-based classification [28], in this section, we introduce a new taxonomy for combination rules, emphasizing conflict management strategies and cardinality integration. When the conflict between sources reaches a value of 1, indicating complete disagreement, traditional rules like DS are observed to be ineffective in combining the

[3] The power set of any set S, written $\mathcal{P}(S)$ or 2^S, is the set of all subsets of S, including the empty set and S itself.

[4] Under closed-world assumption $m(\varnothing) = 0$. Under open-world assumption, $m(\varnothing)$ may be positive.

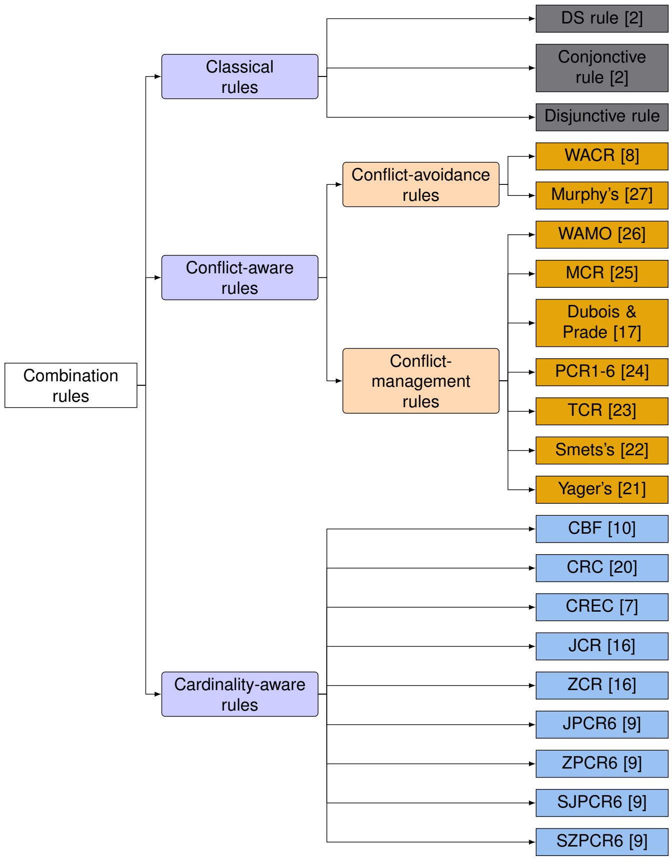

Fig. 1. Combination rules taxonomy.

masses. This limitation has prompted the development of a new category of rules known as *conflict-aware* combination rules. These rules can be categorized into two main groups: *conflict-avoidance* and *conflict-management*.

Conflict-avoidance rules address conflicts before combining evidence through preprocessing techniques, modifying the original BBAs before their combination. These rules can be further divided into two categories: those that utilize the DS rule and those that do not. For instance, Murphy [27] employs arithmetic averaging on the evidence before applying the DS rule. This additional categorization is not depicted in Fig. 1 for simplicity reasons.

In contrast to conflict-avoidance rules, conflict-management rules focus on redistributing conflicts among bodies of evidence. These rules can be further categorized into those redistributing the total conflict and those redistributing the partial conflict; however, this additional categorization is not depicted in Fig. 1 for simplicity. For instance, Smets [22] assign total conflicting mass to the empty set, while Yager [21] assigns it to the universal set Θ. Dezert and Smarandache [24] redistribute partially conflicting belief masses among specific elements of the powerset involved in the conflicting masses. Specifically, the Proportional Conflict Redistribution rule (PCR6) with ($s > 2$) sources of evidence is given by [9]:

$$m_{1,\ldots,s}^{PCR6}(H) = m_{1,\ldots,s}(H) + CR^{PCR6}(H) \qquad (7)$$

where $m_{1,\ldots,s}(H)$ represents the conjunctive consensus on H between the s sources, and $CR^{PCR6}(H)$ is the part of the conflicting masses redistributed back to the elements involved in the conflict and proportionally to their original masses.

Conflict-aware rules, due to their additional computations, often require more computational resources compared to classical rules. This increased demand for processing power and time is particularly notable because both conflict-avoidance and conflict-management rules play a crucial role in enhancing the adaptability of BFT to real-world situations, where conflicting evidence is often inherent and demands careful consideration for more accurate and reliable decision outcomes.

Recently, a new category of combination rules has emerged, known as *cardinality-aware* rules. These rules incorporate the consideration of the cardinality of focal set elements, aiming to further enhance the decision-making process [7]–[10], [16], [20], [23]. Conflict- and cardinality-aware categories are not mutually exclusive. This implies that a rule, such as the CBF rule [10], can be both conflict- and cardinality-aware. In our proposed taxonomy (Fig. 1), if a combination rule is cardinality-aware, it is categorized as such, regardless of its conflict-awareness aspect.

To grasp the heightened complexity of cardinality-aware rules, let us examine the CBF rule [10], which involves two steps. In the first one, the original BBA is transformed into a modified BBA, denoted as m_*, where a pignistic probability transformation (Eq. 6) is carried out for each singleton focal element (α_i). The belief values associated with multi-subset propositions (β_j) are set to zero as follows:

$$\begin{aligned} m_*(\alpha_i) &= \mathrm{BetP}_m(\alpha_i), \quad |\alpha_i| = 1 \\ m_*(\beta_j) &= 0, \quad |\beta_j| > 1 \end{aligned}$$

In the second step, belief from a single subset proposition (α_i) is transferred to multi-subset propositions (β_j), where $\alpha_i \subset \beta_j$. The outcome of this transfer is termed the "transferred BBA", denoted as m_{**}, in which each single subset proposition needs to transfer a part of its belief to a correlated multi-subset proposition according to Equation (8).

$$m_{**}(\alpha_i) = m_*(\alpha_i) - \sum_{\alpha_i \cap \beta_j \neq \emptyset} \frac{m_*(\alpha_i)}{(2^n - 1) \cdot |\beta_j|} \qquad (8)$$

Consequently, each multi-subset proposition receives belief from a single subset proposition according to Equation (9).

$$m_{**}(\beta_j) = \sum_{\alpha_i \cap \beta_j \neq \emptyset} \frac{m_*(\alpha_i)}{(2^n - 1) \cdot |\beta_j|} \qquad (9)$$

Finally, the resulting transferred BBAs (Equations 8 and 9) are combined using the DS rule.

B. Discussion

From the previous section, it is evident that cardinality-aware combination rules demonstrate a higher level of computational complexity compared to classical and conflict-aware rules. A further challenge arises from the fact that existing frameworks, such as the one developed in [14], lack mechanisms to effectively handle or integrate the cardinality aspect in combination rules. This leads to highly inefficient implementations of these rules, severely limiting their adoption in practical applications. Furthermore, the absence of a unified implementation makes it difficult to comprehensively assess and compare the performance of various cardinality-aware rules. To overcome the limitations above, this paper aims not to create an entirely new framework but to enhance the existing one [14] by introducing new optimized functions. This augmentation is intended to facilitate seamless implementations and assessments of both existing and novel cardinality-aware combination rules.

IV. Our Proposal

This section introduces the improved framework and explains how it tackles the challenges we have identified earlier. It also outlines the framework's contribution to effectively implementing cardinality-aware rules in BFT.

A. High-level framework architecture

Figure 2 illustrates the high-level architecture of the proposed framework. It consists of five layers, each serving specific functions. The first layer, highlighted in Sec. IV-B, is the power set, considered as the foundation of the framework. The second layer incorporates optimized key functions essential for efficiently implementing both existing and new combination rules, including cardinality-aware ones. Detailed discussion on representative key functions will follow. The third layer comprises all combination rules, while the fourth layer facilitates decision-making based on various criteria. Finally, the application layer utilizes the combination and decision layers to accommodate a wide range of practical applications.

B. Representation of the powerset 2^Θ

Unlike probabilities, which are solely defined on Θ, BFT relies on functions defined on the powerset 2^Θ. Therefore, the encoding of 2^Θ significantly influences the computational complexities of all functions. To tackle this issue, the previous implementation [14] utilized a vector representation. To maintain complete compatibility, we also adopt the same representation, which is structured as follows:

- **Binary FoD**: When the FoD comprises only two elements $\{\theta_1, \theta_2\}$, the MATLAB vector is constructed as $[\emptyset, \theta_1, \theta_2, \theta_1 \cup \theta_2]$.
- **Expansion for larger FoD**: For frames containing more than two elements, the vector is expanded iteratively. To illustrate, consider a FoD containing three elements $\{\theta_1, \theta_2, \theta_3\}$. The MATLAB vector expands as follows:
  1) Begin with the foundational vector: $[\emptyset, \theta_1, \theta_2, \theta_1 \cup \theta_2]$.
  2) Concatenate this vector with a new vector formed by combining each element of the frame with the third element: $[\theta_3, \theta_1 \cup \theta_3, \theta_2 \cup \theta_3, \theta_1 \cup \theta_2 \cup \theta_3]$.
  3) The final representation of 2^Θ becomes: $[\emptyset, \theta_1, \theta_2, \theta_1 \cup \theta_2, \theta_3, \theta_1 \cup \theta_3, \theta_2 \cup \theta_3, \theta_1 \cup \theta_2 \cup \theta_3]$.

Employing the aforementioned power set representation, the new framework introduces a refined methodology for managing mass functions. To illustrate, let us consider the following example with $\Theta = \{a, b, c\}$.

$$m_1(a) = 0.5, \quad m_1(b) = 0.2, \qquad m_1(a,b,c) = 0.3$$
$$m_2(a) = 0.5, \quad m_2(a,b) = 0.1, \qquad m_2(c) = 0.4$$

The MATLAB code corresponding to express the two masses according to the original framework is as follows:

```
m1 = [0 0.5 0.2 0    0    0 0 0.3]
m2 = [0 0.5 0   0.1 0.4 0 0 0  ]
```

In the updated implementation, alongside the existing representation, the masses can also be articulated more intuitively before their mapping into vector representation:

```
m1 = 'a(0.5)  b(0.2) abc(0.3)'
m2 = 'a(0.5)  ab(0.1)  c(0.4)'
```

C. Key Functions

In the following subsections, we highlight specific newly introduced key functions, corresponding to the components depicted in the second layer of Fig. 2.

1) $get\_Cardinality(focal)$: This function is a recursive algorithm devised to compute the cardinality of a provided focal element. In this context, the cardinality of a focal element refers to the count of elements it encompasses.

```
function [Cardinality] = get_Cardinality(focal)
    % Is the focal element the empty set?
    if(focal == 1)
        Cardinality = 0;
        return
    % Is it one of the singletons?
    elseif ((focal == 2) || (focal == 3))
        Cardinality = 1;
        return
    % Is it the union of two singletons?
    elseif (focal == 4)
        Cardinality = 2;
        return
    end

    % It is not the base case
    res = focal;
    iteration_count = 0; % number of iterations
    while (res > 2)
        res = res / 2;
        iteration_count = iteration_count + 1;
    end
    frame_size = iteration_count + 1; % size of FoD
    offset = focal - 2^(frame_size - 1);
    Cardinality = 1 + get_cardinality(offset);
end
```

The function first checks specific cases for focal elements $\emptyset$, θ_1, θ_2, and $\theta_1 \cup \theta_2$. In these cases, the cardinality is assigned predetermined values ([**0, 1, 1, 2**]). For other focal elements, the function calculates the cardinality recursively by determining the offset relative to the FoD and iteratively dividing the

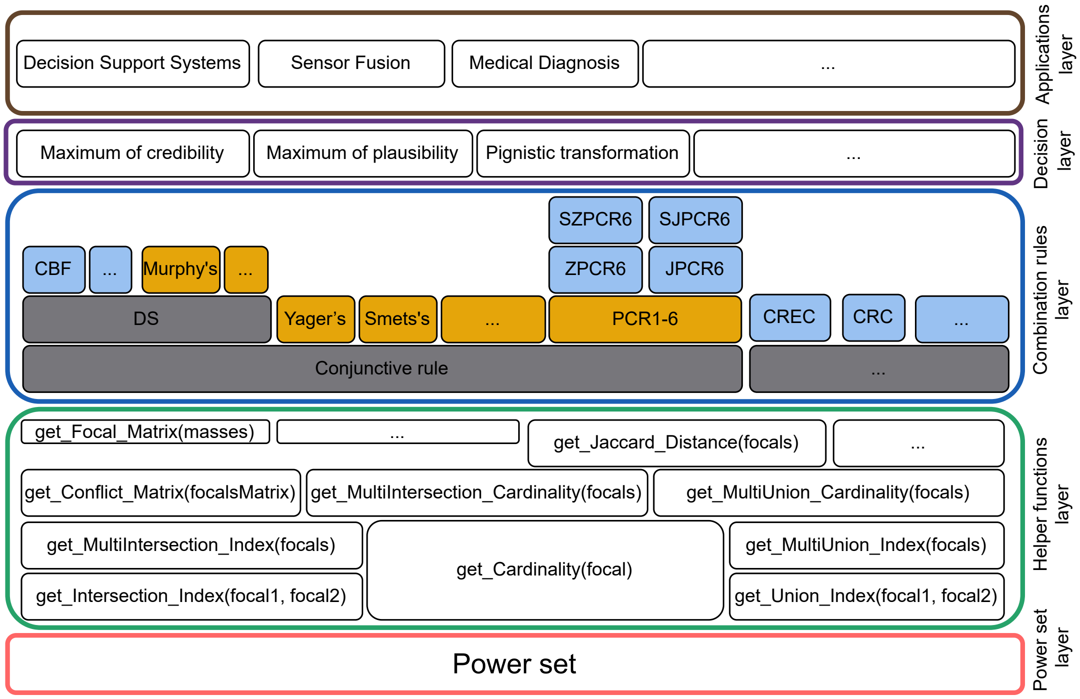

Fig. 2. High-level architecture of the proposed framework.

focal element until reaching a base case. For instance, the following vector: $[\emptyset, \theta_1, \theta_2, \theta_1 \cup \theta_2, \theta_3, \theta_1 \cup \theta_3, \theta_2 \cup \theta_3, \theta_1 \cup \theta_2 \cup \theta_3]$ yields the associated cardinalities: $[\mathbf{0}, \mathbf{1}, \mathbf{1}, \mathbf{2}, 1, 2, 2, 3]$.

2) $get\_Intersection\_Index(focal1, focal2)$: This function computes the intersection between two focal elements and returns the index of the resultant focal element. Similar to the preceding key function, a recursive strategy is utilized to determine the intersection index, taking into account the structure of the powerset vector. The base case occurs when both focal elements are among the first four elements of the powerset and is defined as follows:

```
if (focal1 <= 4) && (focal2 <= 4)
    % Base case matrix
    Mat_Intersection = [1 1 1 1
                        1 2 1 2
                        1 1 3 3
                        1 2 3 4]
    % Get the intersection index from the matrix
    Index = Mat_Intersection(focal1, focal2);
    return
```

The values at the intersection of row i and column j represent the resulting index when $focal1$ and $focal2$ are intersected.

3) $get\_Union\_Index(focal1, focal2)$: The function computes the union between two focal elements and returns the index of the resultant focal element. Employing recursion, it determines the union index by considering focal elements' cardinality and relationship within the powerset vector. The base case applies when both focal elements are within the first four elements of the powerset and is defined as follows:

```
if (focal1 <= 4) && (focal2 <= 4)
    % Base case matrix
    Mat_Union = [1 2 3 4
                 2 2 4 4
                 3 4 3 4
                 4 4 4 4]
    % Get the union index from the matrix
    Index = Mat_Union(focal1, focal2);
    return
```

The entries at the intersection of row i and column j denote the resulting index of the union of $focal1$ and $focal2$.

4) Sequential Intersection and Union Operations: The preceding functions, $get\_Intersection\_Index(focal1, focal2)$ and $get\_Union\_Index(focal1, focal2)$, facilitate the computation of intersection and union operations for pairs of focal elements. Leveraging the associativity property of these operations, they enable sequential computation for a vector of focal elements. Consequently, through iterative application, multiple focal elements' intersection or union can be efficiently computed. Building on this capability, two additional functions have been developed: $get\_MultiIntersection\_Index(focals)$ and $get\_MultiUnion\_Index(focals)$. Moreover, two functions to calculate the resulting focal element's cardinality after performing intersection or union operations have been devised, namely: $get\_MultiIntersection\_Cardinality(focals)$ and $get\_MultiUnion\_Cardinality(focals)$.

5) $get\_Conflict\_Matrix(Focal\_Matrix, tab\_conflict)$: This function plays a vital role in analyzing conflict within BFT by systematically exploring all potential combinations of partial conflicts among multiple sources. Given a matrix of focal elements, it meticulously assesses each source's contribution and detects intersecting focal elements where conflicts emerge. For instance, in a scenario involving multiple sources, $get\_Conflict\_Matrix$ calculates a conflict matrix, elucidating the intersections between focal elements linked to conflicting masses.

Before computing the conflict matrix, we construct an optimized representation of the masses to exclude the null hypotheses from the computation. This task is managed by the $get\_Focal\_Matrix(Masses)$ function, which yields a matrix containing only focal element indices. To exemplify, let us revisit the previous example (Section IV-B). The resulting focal matrix is:

```
Focal_Matrix = [2  2
                3  4
                8  5]
```

$$\Leftrightarrow \quad \begin{matrix} a & a \\ b & ab \\ abc & c \end{matrix}$$

Using the focal matrix provided earlier, the $get\_Conflict\_Matrix$ function yields the associated $Conflict\_Matrix$:

```
Conflict_Matrix = [2  5
                   3  2
                   3  5]
```

$$\Leftrightarrow \quad \begin{matrix} a & c \\ b & a \\ b & c \end{matrix}$$

Following the introduction of the most important key functions in our new framework, the subsequent subsection showcases the re-implementation of the PCR6 combination rule for illustrative purposes.

D. Example: PCR6 implementation revisited

The original implementation of the complicated classical combination rule PCR6 becomes untractable for a large number of elements in Θ. To assess the efficacy of our proposed functions, we demonstrate how the new implementation of PCR6 utilizes the newly developed functions as illustrated in the following MATLAB code segment:

1) Conjunction calculation (line 6): The initial step involves calculating the conjunctive masses. While the implementation employs the original DST function provided by the framework for this purpose, it integrates seamlessly with the subsequent steps, laying the foundation for further computation.
2) Conflict matrix computation (lines 9-10): In this step, the implementation harnesses the power of the get_Focal_Matrix and $get\_Conflict\_Matrix$ functions (Sec. IV-C5) to compute the conflict matrix. This matrix serves as a crucial tool in identifying the intersections between focal elements associated with conflicting masses, providing invaluable insights into the sources of conflict within BFT.
3) Weights calculation (lines 14-21): With the conflict matrix in hand, the implementation calculates and normalizes the weight for each focal element based on the masses assigned to conflicting hypotheses. This iterative process ensures a thorough examination of conflicting sources and their respective contributions, enabling precise determination of weights.
4) Partial conflicts distribution (lines 23-26): Using the previously computed weights and conflict matrix, this step distributes the partial conflicting masses among the focal elements involved in the conflicts.
5) Combination (line 28): The final masses for the PCR6 combination rule are derived by aggregating the conjunctive masses with the conflicts.

```
1 function [Combined_Masse]=PCR6_Optimized(MasseIn)
2 % Size of the power set and the number of sources
3 POWER_SET_SIZE = size(MasseIn, 1)
4 SOURCES_NUMBER = size(MasseIn, 2)
5 % Calculate conjunctive masses
6 Conj_masses = get_Conjunctive_Masses(MasseIn)
7 Conflict_Masse = zeros(1, POWER_SET_SIZE)
8 % Calculate the conflict matrix
9 Focal_Mat = get_Focal_Matrix(MasseIn)
10 Conflict_Matrix = get_Conflict_Matrix(Focal_Mat, [])
11 % Get the number of partial conflicts
12 [partial_conflict_nb, ~] = size(Conflict_Matrix)
13 % Compute the weights for each partial conflict
14 for k=1: partial_conflict_nb
15     weight = zeros(1, POWER_SET_SIZE)
16     for i = 1:SOURCES_NUMBER
17         weight(Conflict_Matrix(k, i)) =
               weight(Conflict_Matrix(k, i)) +
               MasseIn(Conflict_Matrix(k, i), i)
18     end
19     % Normalize the weights
20     lambda = sum(weight)
21     normalized_weight = weight / lambda
22     % Distribute partial conflicting masses using
         the computed weights
23     for i = 1:length(weight)
24         Conflict_Masse(i) = Conflict_Masse(i) +
               normalized_weight(i) *
               get_Conjunctive_Mass(MasseIn,
               Conflict_Matrix(k, :))
25     end
26 end
27 % Add conjunctive masses and return combined masses
28 Combined_Masse =  Conj_masses + Conflict_Masse
```

In summary, the aforementioned implementation showcases the successful integration of original DST functions with newly developed tools, facilitating the efficient computation of the PCR6 combination rule within BFT. The computational complexity of the new PCR6 implementation will be further evaluated in the subsequent section, contrasting it with the original approach.

E. Combination, Decision, and Applications layers

The new framework guarantees backward compatibility with the original version while incorporating recent cardinality-aware rules such as CFB [10], ZCR [16], JCR [16], and four PCR6 variants [9]: ZPCR6, JPCR6, SZPCR6, and SJPCR6.

The decision layer incorporates commonly employed decision criteria such as maximum belief, maximum plausibility, and the Pignistic transformation. These decision rules empower BFT to effectively manage uncertainty and conflict in various applications implemented at the application layer.

V. Performance Evaluation

A. Experimental setup

In this performance evaluation, two experiments are conducted. The first one focuses on complicated classical combination rules, such as PCR6, intending to demonstrate the advantages of using the enhanced framework across diverse scenarios. The second experiment assesses the complexity of newly added cardinality-aware rules, including PCR6 variants: ZPCR6, JPCR6, SZPCR6, and SJPCR6. These rules are not available in the original framework. The experiments utilized a PC with an Intel Core i5-4340M CPU @ 2.90GHz, 4GB RAM, running Windows 8.1 Pro (64-bit), and MATLAB R2015a.

B. Results and discussions

In the first experiment, we assess the computational complexity of the original PCR6 implementation compared to the optimized version. The assessment involves varying the size of the FoD $\Theta = \{h_i,\ i = 1...n\}$ and observing the corresponding execution times for both implementations. Two belief assignments are taken into consideration:

$$m_1(h_1) = 0.5 \qquad m_1(\Theta \setminus \{h_1\}) = 0.5$$
$$m_2(h_2) = 0.5 \qquad m_2(h_1 \cup h_3) = 0.5$$

Table I shows the obtained results. The original PCR6 exhibits exponential growth as the $|\Theta|$ increases, whereas the new implementation's running time does not significantly change with an increase in $|\Theta|$. Additionally, as shown in Fig. 3, the new implementation consistently outperforms the original PCR6 across different frame sizes. This highlights the efficiency gains achieved through the use of newly developed functions.

TABLE I
Execution time of PCR6 while varying the size of Θ

	Execution time of PCR6 (s)	
$\lvert\Theta\rvert$	original implementation	new implementation
3	0.0049	0.00048
4	0.0198	0.00049
5	0.0829	0.00050
6	0.3063	0.00059
7	1.1794	0.00075
8	4.6171	0.00110
9	18.0928	0.00170
10	72.0245	0.00270
11	290.9220	0.00480

Furthermore, we investigate the combination complexity by varying the number of sources while keeping the size of the FoD fixed at 4. We randomly select 4 focal elements with non-zero mass. The results are summarized in Fig. 4. The enhanced implementation of the PCR6 rule consistently shows significantly reduced execution times across various numbers of sources.

Figures 3 and 4 reveal that the enhanced implementation consistently outperforms the original, showcasing superior performance across diverse frame sizes and numbers of sources.

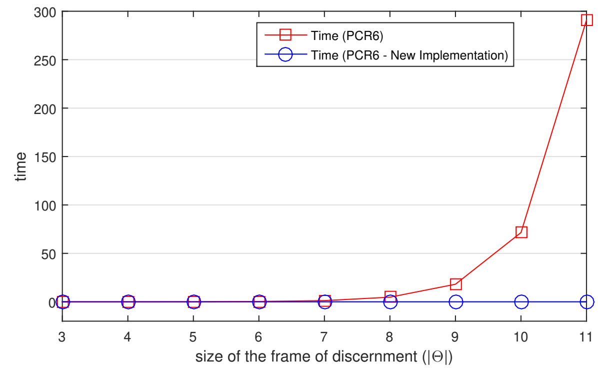

Fig. 3. Execution time of PCR6 while varying the size of Θ.

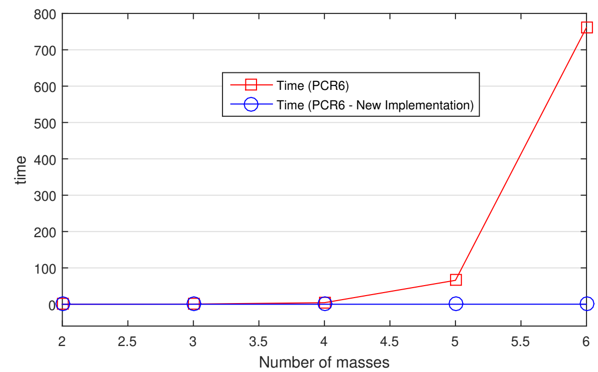

Fig. 4. Execution time of PCR6 with varied numbers of sources.

Consequently, it stands out as the preferred choice for efficient BFT combinations in various applications.

In this second experiment, we have integrated the four PCR6 variants (ZPCR6, JPCR6, SZPCR6, and SJPCR6) into the new framework. This process included the incorporation of Zhang and Jaccard's distances. Figures 5 and 6 depict the execution times for the four variants across different Θ sizes and numbers of sources, respectively.

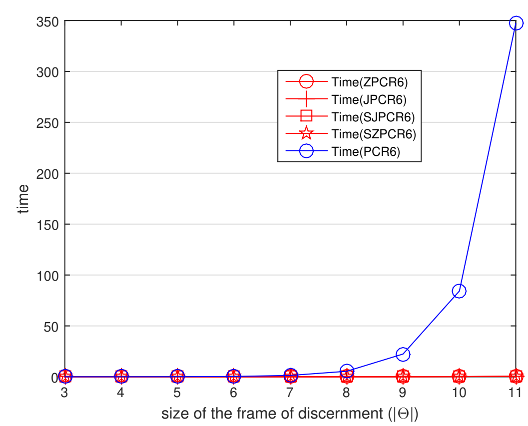

Fig. 5. Execution time of ZPCR6, JPCR6, SZPCR6, SJPCR6, and PCR6.

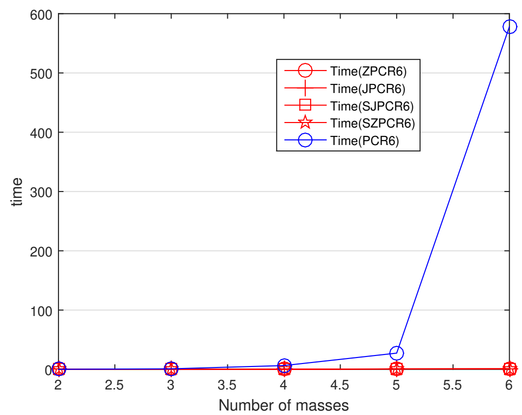

Fig. 6. Execution time of ZPCR6, JPCR6, SZPCR6, SJPCR6, and PCR6.

As evident from Fig. 5, the execution time does not exceed 0.5 seconds, while it reaches 350 seconds in the original PCR6 implementation. Similarly, in Fig. 6, the execution time remains below 2 seconds when the number of sources reaches 6, in contrast to the original PCR6 where the execution time exceeds 580 seconds. These observations underscore once more the importance of the newly implemented functions.

VI. Concluding Remarks

This paper introduces an enhanced belief functions framework. Various key functions were designed and implemented to reduce the computational complexity of both complex classical combination rules (PCR6) and recent cardinality-aware rules (ZCR, JCR, CBF, and the four PCR6 variants). Experimental results confirmed the efficiency of the proposed framework, demonstrating improved performance in handling both conflict-aware and cardinality-aware belief combination scenarios. This improvement is intended to enhance the usability of BFT in practical applications. The newly developed framework is now accessible to the community, offering opportunities for utilization and potential further enhancements.

In future work, we plan to investigate further the codification of the powerset as it influences the whole complexity of the framework. Furthermore, we aim to expand the code base by incorporating additional and more recent combination rules.

References

[1] G. Shafer, *A mathematical theory of evidence*. Princeton university press, 1976, vol. 42.

[2] A. P. Dempster, "Upper and lower probabilities induced by a multivalued mapping," in *Classic works of the Dempster-Shafer theory of belief functions*. Springer, 2008, pp. 57–72.

[3] M. R. Senouci, A. Mellouk, M. A. Senouci, and L. Oukhellou, "Belief functions in telecommunications and network technologies: an overview," *annals of telecommunications-annales des télécommunications*, vol. 69, no. 3, pp. 135–145, 2014.

[4] N. Chibani, F. Sebbak, W. Cherifi, M. R. Senouci, and S. Ouchani, "An uncertainty-aware road anomalies detection based on dempster–shafer theory," in *First International Conference on Computer Communications and Intelligent Systems (I3CIS)*. IEEE, 2022, pp. 25–30.

[5] F. Sebbak, F. Benhammadi, A. Chibani, Y. Amirat, and A. Mokhtari, "Dempster–shafer theory-based human activity recognition in smart home environments," *annals of telecommunications-annales des télécommunications*, vol. 69, no. 3-4, pp. 171–184, 2014.

[6] K. Sentz and S. Ferson, "Combination of evidence in dempster-shafer theory." Sandia National Laboratories, 2002.

[7] F. Sebbak, F. Benhammadi, M. Mataoui, S. Bouznad, and Y. Amirat, "An alternative combination rule for evidential reasoning," in *the 17th International Conference on Information Fusion (FUSION 2014)*. IEEE, 2014, pp. 1–8.

[8] F. Sebbak, S. Bouznad, F. Benhammadi, A. Chibani, and Y. Amirat, "Context awareness in uncertain pervasive computing and sensors environment," in *2018 21st International Conference on Information Fusion (FUSION)*. IEEE, 2018, pp. 1–5.

[9] F. Smarandache and J. Dezert, "Modified pcr rules of combination with degrees of intersections," in *Information Fusion (Fusion), 2015 18th International Conference on*. IEEE, 2015, pp. 2100–2107.

[10] Y. Tang, X. Zhang, Y. Zhou, Y. Huang, and D. Zhou, "A new correlation belief function in dempster-shafer evidence theory and its application in classification," *Scientific Reports*, vol. 13, no. 1, May 2023.

[11] P. Smets, "Softwares for the TBM," http://iridia.ulb.ac.be/ psmets/#G, 2004, [Accessed 10-03-2024].

[12] T. Denoeux, "Approximation of belief functions," https://www.hds.utc.fr/ tdenoeux/dokuwiki/doku.php?id=en:software, 2000, [Accessed 10-03-2024].

[13] T. Reineking, "A python library for performing calculations in the dempster-shafer theory of evidence," http://reineking.github.io/pyds/, 2021, [Accessed 10-03-2024].

[14] A. Martin, "Implementing general belief function framework with a practical codification for low complexity," in *Advances and Applications of DSmT for Information Fusion, (Collected Works, Vol 3), F. Smarandache and J. Dezert (Editors)*. American Research Press Rehoboth, 2009, pp. 217–273.

[15] R. R. Yager, "On the fusion of imprecise uncertainty measures using belief structures," *Information Sciences*, vol. 181, no. 15, pp. 3199–3209, 2011.

[16] L. Zhang, "Representation, independence, and combination of evidence in the dempster–shafer theory," in *Advances in the Dempster–Shafer theory of evidence*. John Wiley & Sons, Inc., 1994, pp. 51–69.

[17] D. Dubois and H. Prade, "Representation and combination of uncertainty with belief functions and possibility measures," *Computational intelligence*, vol. 4, no. 3, pp. 244–264, 1988.

[18] Z. Elouedi, K. Mellouli, and P. Smets, "Assessing sensor reliability for multisensor data fusion within the transferable belief model," *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 1, p. 782–787, Feb. 2004.

[19] P. Smets and R. Kennes, "The transferable belief model," *Artificial intelligence*, vol. 66, no. 2, pp. 191–234, 1994.

[20] F. Sebbak, F. Benhammadi, S. Bouznad, A. Chibani, and Y. Amirat, "An evidential fusion rule for ambient intelligence for activity recognition," in *BELIEF 2014, LNAI 8764*. Springer, 2014, pp. 356–364.

[21] R. R. Yager, "On the dempster–shafer framework and new combination rules," *Information sciences*, vol. 41, no. 2, pp. 93–137, 1987.

[22] P. Smets, "The combination of evidence in the transferable belief model," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 12, no. 5, pp. 447–458, 1990.

[23] F. Sebbak and F. Benhammadi, "Total conflict redistribution rule for evidential fusion," in *Information Fusion (FUSION), 2016 19th International Conference on*. IEEE, 2016, pp. 1324–1331.

[24] M. Arnaud and O. Christophe, "Toward a combination rule to deal with partial conflict and specificity in belief functions theory," in *10th International Conference on Information Fusion, 2007*. IEEE, 2007, pp. 1–8.

[25] F. Sebbak and F. Benhammadi, "Majority-consensus fusion approach for elderly iot-based healthcare applications," *Annals of Telecommunications*, vol. 72, no. 3-4, pp. 157–171, 2017.

[26] F. Sebbak, F. Benhammadi, A. Chibani, Y. Amirat, and A. Mokhtari, "New evidence combination rules for activity recognition in smart home," in *the 16th International Conference on Information Fusion (FUSION 2013), 2013*. IEEE, 2013, pp. 46–52.

[27] C. K. Murphy, "Combining belief functions when evidence conflicts," *Decision support systems*, vol. 29, no. 1, pp. 1–9, 2000.

[28] P. Smets, "Analyzing the combination of conflicting belief functions," *Information Fusion*, vol. 8, no. 4, pp. 387–412, 2007.