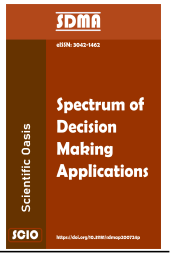
SCIENTIFIC OASIS

Spectrum of Decision Making and Applications

Journal homepage: www.dmap-journal.org
ISSN: 3042-1462



Pythagorean Soft Sets and Hypersoft Sets: A Comprehensive Framework for Advanced Uncertainty Modeling in Decision

Muhammad Tahir[1], Muhammad Kamran[2,*], Misbah Rasheed[3], Abdul Hanan[4], Muhammad Imran Shahid[5]

[1] Department of Mathematics, Institute of Numerical Sciences, Gomal University, Dera Ismail Khan, 29050, KPK, Pakistan
[2] Research Institute of Business Analytics and SCM, College of Management, Shenzhen University, China
[3] Institute of Mathematics, Khwaja Fareed University of Engineering and Information Technology, Rahim Yar Khan, Pakistan
[4] Faculty of Physical and Mathematical Sciences, The Islamia University of Bahawalpur, Pakistan
[5] Department of Mathematics and Statistics, University of Southern Punjab Multan, Pakistan

ARTICLE INFO

Article history:
Received 2 July 2025
Received in revised form30 August 2025
Accepted 27 September 2025
Available online 4 October 2025

Keywords:
Pythagorean Fuzzy Set; Soft Set; Hypersoft Set; Uncertainty Modeling; Decision-Making

ABSTRACT

This paper introduces a comprehensive mathematical framework that integrates Pythagorean fuzzy sets (PyFS), soft sets (SS), and hypersoft set theory to establish robust methodologies for addressing complex uncertainties in multi-attribute decision-making (MADM). We present two formal approaches: Pythagorean soft sets (PySS) and Pythagorean hypersoft sets (PyHSS), which enhance the conventional soft set and hypersoft set frameworks by incorporating the expressiveness of Pythagorean fuzzy sets (PyFS). The theoretical framework demonstrates that these structures preserve all fundamental set-theoretic operations and facilitate the representation of the membership degree (MD) and non-membership degree (n-MD) with sums not exceeding 1, contingent upon satisfying the Pythagorean condition $\psi^2 + \phi^2 \leq 1$. We illustrate the applicability of our approach through extensive digital implementations in technology selection, cloud-based configuration, and educational recommendation systems. The mathematical functionalities exhibited, including operation closure and generalisation hierarchies, render PySS and PyHSS formidable decision support system instruments for managing imprecise and ambiguous information across several criteria.

*Corresponding author.
E-mail address: kamrankfueit@gmail.com

https://doi.org/10.31181/sdmap41202761

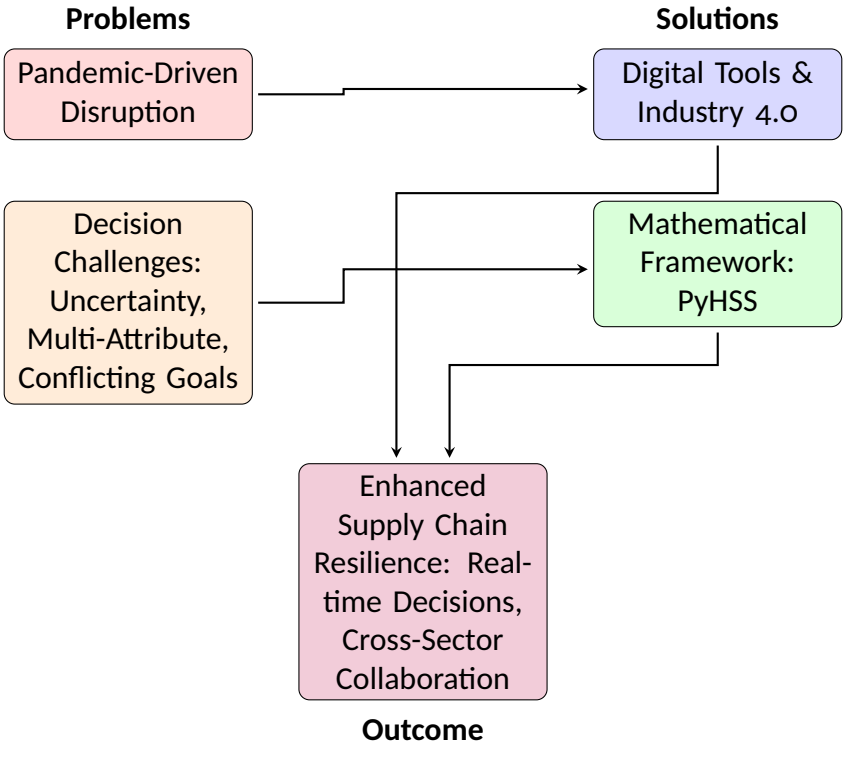
1. Introduction

The COVID-19 pandemic introduced a historic shift in global supply chain management that highlighted how the traditional linear supply chains were fatally flawed and accelerated the process of switching to digital technologies [1]. It states that the pandemic brought some industries into chaos by putting them out of business or exposing them to demand that they could not meet, as well as by disrupting supply chains, forcing them to make changes to operations management that they had earlier deemed impractical [2]. This turmoil revealed the insensitivity to establishing collaborative integrated strategies of production management that extend beyond the company to whole network ecosystems. It resulted in enormous downstream consequences, with even the weak indicators of supply problems (minor material availability, increased delivery time) already reaching China in the final months of 2019, not always being processed by some companies due to the absence of coordination and timely information sharing, and primarily because of the first pandemic epicenters, such as China [3]. The foregoing empirical findings render the necessity for the immediate availability of digital tools that could help achieve real-time sharing of information, monitor the entire supply chain, and formulate adaptive strategies to suit the specific needs of the market in the context of a crisis very significant [4]. Supply chain digital transformation represents a paradigm shift from traditional linear supply chains to integrated and intelligent networks that can dynamically react to disruption [4]. Industry 4.0 is disruptive because it restructures the organizations and management of industrial value networks by concertedly applying disruptive digital technologies like augmented reality, artificial intelligence (AI), big data analytics, cloud computing, blockchain, the internet of things (IoT), robotic systems, and simulation tools to the value networks [5]. This digital ecosystem has enabled the fundamental design principles of interoperability, real-time operation, modularity, decentralization, and integrability, which are essential for realizing the concept of a hyperconnected value creation system. Within this context, our paper introduces Pythagorean soft sets (PySS) [6] and Pythagorean hypersoft sets (PyHSS) [7] as the next stages of mathematical modeling to aid in digital supply chain decision-making in cases where complexity, uncertainty, and multi-attribute issues prevail in the supply network environment.

Modern supply chains are confronted with a convergence of risk factors in a way never experienced before and are challenging to risk management methods [8]. According to the records, the supply chain disruptions that businesses have experienced lately have no precedent in the recent history of the world economy, with the COVID-19 pandemic, radical shifts in the global consumer markets, and geopolitical changes being the leading drivers of external disruption. Such shocks have hit areas of the economy such as “advanced semiconductor chip production, medical products, and energy provision and distribution,” exposing systemic vulnerabilities in over-efficient yet brittle supply chains [9]. According to the experts, the majority of modern-day supply chains have been overemphasizing productivity and efficiency and neglecting resilience, leaving operational efficiency and disruption resilience out of balance. The pandemic especially emphasized the need to work across sectors and production repurposing capabilities. Companies experienced the necessity to reuse production lines to cover shortages in products such as masks and other personal protective gear, ventilators and other clinical care equipment, and new partnerships with government and other state institutions such as schools and hospitals. At the same time, machine builders and long supply chain vendors were grappling with services such as maintenance and commissioning of machines and reconfiguring machines to support new processes, which they say could only be experienced by digitized companies [10]. Consumer goods sectors were compelled to establish an entirely online business model and had to reorganize their activities, both in the production reconfiguration and in the supply and distribution reconfiguration, to access materials and provide products that triggered specific channels with suppliers and customers. These compound risks require complex mathematical frameworks that can manage

these multi-faceted, multi-attribute decision-making (MADM) environments that define the present supply chain ecosystems.

Modern supply chain decision-making is complex due to the interactions of many factors: uncertainty in demand and supply, conflicting objectives among stakeholders, dynamic risk profiles, and demands to adapt in real time [11]. Conventional decision-making models have a hard time with the beyond-worst-case scenarios that typify pandemic-like disruptions, in which the operating environment has extreme uncertainty and ambiguity, and traditional probability distributions cannot capture this extreme uncertainty and ambiguity. The Pythagorean model presented in this paper overcomes these problems by offering a mathematical framework capable of representing cases where membership degree (MD) and non-membership degree (n-MD) are additive to more than 1 whilst meeting the Pythagorean constraint of in-representation that $\psi^2 + \phi^2 \leq 1$ and thus offers the ability to represent more complex uncertainty patterns than a fuzzy or intuitionistic fuzzy approach. Disruption response activities in the supply chain require the trade-off of various competing goals, including continuity of operations, staff security, financial sustainability, and social accountability. These goals are usually incompatible, typically forming decision environments in which satisfactory resolutions have to make trade-offs between efficiency and resilience, cost and responsiveness, and short-term survival and long-term viability. The multi-dimensionality of such decisions, such as reliability of the supplier, flexibility of transportation, flexibility of inventory, and flexibility of production, necessitates mathematical frameworks that are capable of considering many parameters at once, each with many interactions between them. This is exactly what our PySS and PyHSS offer: the ability to analyze the results of multi-attribute decisions in circumstances of total uncertainty.



It has long been understood that fuzzy logic is a potent tool in managing uncertainty in supply chain management because it makes it possible to have partially true values (between 0 and 1) instead of true/false binary values [12]. But conventional fuzzy sets are limited to MD without n-MD or hesitation aspects [13]. Intuitionistic fuzzy sets (IFS) [14] further generalized this framework by adding both MD (ψ) and n-MD (ϕ) degrees with the restriction that their sum is between 0 and 1, thus limiting

their capacity to model some forms of uncertainty, especially in high-stress settings such as pandemic disruptions where decision-makers may assign both high and high values to MD and n-MD of a given option. This limitation is addressed in Pythagorean fuzzy sets (PyFS) [15] with the more lax condition $\psi^2 + \phi^2 \leq 1$, enabling the representation of situations where the Pythagorean condition is violated but $\psi + \phi > 1$. The IFS example The pair (0.7, 0.7) is invalid in the IFS (0.7 + 0.7 = 1.4 ¿ 1) but valid in the PyFSs ($0.7^2 + 0.7^2 = 0.49 + 0.49 = 0.98 \leq 1$). This increased ability is especially beneficial in making supply chain decisions in times of disruption, when managers may be judging both high suitability and high risk of a specific action, such as switching suppliers or reusing production lines, and require mathematical models capable of reflecting this subtle set of judgments.

Fuzzy logic and its variants provide strong methods to deal with supply chain risk, allowing one to quantify qualitative uncertainty, represent the complexity of interdependence between risk factors, and perform multi-criteria decision-making(MCDM) under ambiguity. Within the framework of the digital supply chain, fuzzy tools could be used to evaluate the effect of different Industry 4.0 technologies on resilience, rank mitigation measures by using numerous conflicting criteria, and allocate resources to the supply network efficiently when uncertainties arise. In particular, fuzzy logic allows (1) to quantify imprecise risk judgments by experts; (2) gradual risk states as opposed to binary risk states (and vice versa); (3) to integrate multiple risk views into single risk assessments; and (4) to reason in the face of incomplete or ambiguous data.These capabilities are augmented by the use of PyFS, which offers a greater problem space in which to encode uncertainty, which is especially helpful when modeling the new risk conditions associated with disruptions of the pandemic nature. To take a specific example, the consideration of the feasibility of repurposing production strategies in the face of COVID-19 was done by considering factors including technical feasibility (high MD), cost implications (high n-MD), timeline to implementation (variable membership/non-membership), and regulatory compliance (variable MD/n-MD). The complex evaluations of supply chain resilience can be better supported by decision support through Pythagorean fuzzy sets compared to traditional methods used to evaluate these decisions.

By introducing our own PySS and PyHSS framework, the benefits of this approach in digital supply chain management are considerable, especially in the face of resilience building under disruptions. Mathematically, our solution has been superior due to the following main properties shown: (1) Enhanced expressive power, via the relaxed Pythagorean condition of being able to accommodate a wider variety of uncertainty patterns; (2) Operational closure under all fundamental set operations (union, intersection, complement, containment) as shown in Theorems; (3) Multi-attribute capability, via the hypersoft set extension of accommodating Cartesian product parameter spaces; and (4) Generalization hierarchy, that traditional soft sets as well as intuitionistic approaches are special cases of the former. In the case of digital supply chain applications in particular, our framework will facilitate (1) uncertainty modeling in real time with complex multi-attribute supply chain situations via PyHSS mappings between attribute tuples and PyFS; (2) adaptive decision support via operations that are mathematically valid in all circumstances; (3) cross-sector collaboration modeling via attribute-restricted operations that filter the set of possible combinations of parameters to particular collaboration situations; and (4) resilience strategy evaluation via weighted aggregation operations that combine multiple resil.

The digital cases in Sections 3 and 2.3 illustrate the feasibility of our method for a real-world digital problem. The smartphone configuration repeater (Example 5) demonstrates how PyHSS can be used to make complex multi-attribute product customization choices, whilst the online course recommendation system (Example 6) demonstrates the use of PyHSS in education supply chains. The case study of cloud service evaluation shows that it can be used to model cross-sector collaboration in supply chain resilience planning. These real-world applications validate our view that our Pythagorean framework has both mathematical and practical merits in solving the complex decision-making chal-

lenges of contemporary digital supply chains. indent The paper contributes to the literature on supply chain resilience in four key ways: (1) it proposes a new mathematical model, PySS and PyHSS, as developed from earlier models to better handle uncertainties; (2) it validates the theoretical properties and operationalism of the new model; (3) it shows how the new model might be applied to digital application decision makers; and (4) it provides algorithmic and visual models to implement the new model. These contributions are developed in the subsequent sections in a systematic manner, starting with basic concepts and then moving towards theoretical basis, operational frameworks, examples of applications, and implementation guidelines.

1.1 Literature Review

Mathematical modeling of uncertainty has developed far beyond its classical roots to reflect the complexities of modern decision-making issues [16], with precision yet little or no flexibility to respond to the imprecision of the real world. In response, we introduced the principle of the so-called fuzzy sets (FS)[17], which possess a partial membership and can be used to model the gradedness and imprecision to some extent, up to a given degree of attention. The IFS has furthered this by introducing, in addition to the MD, an n-MD to better explain the hesitation of an expert [14]. The need to be able to model even more complex uncertainty, where MD and n-MD can add to more than one, led to the introduction of the so-called PyFS [15], which no longer demands that the sum of MD and n-MD add to one, expanding considerably the scope of uncertainty that can be modeled. Parallel evolutionary direction A parallel evolutionary direction on parameterized uncertainty was introduced as the strong parameterized tool, whereby the elements of a set of parameters map to a subset of a universal set as a structured way of working with attribute-based ambiguity [18]. This gave rise to the extension of the notion of soft sets [19] into so-called hypersoft sets [20], in which the mappings are defined on the Cartesian product of several sets of attributes that are not in general similar, to allow the modelling of multi-parametric effects. It was also generalized to the concept of super hypersets [21] in order to give the most optimal parametric interdependency, and to give an unparalleled model of complex, multi-dimensional uncertainty by mapping subsets of attribute sets to the universal set [22]. It was desirable to provide an unparalleled model of complex, multi-dimensional uncertainty in this way. These two evolutionary directions combined a fuzzy extension and parameterized structure, yielded even more powerful hybrids: the parameterization of soft sets and the more powerful uncertainty representation of PyFS, called PySS [6], to give them their name. It was subsequently expanded to PyHSS [7], which combines the expressive power of PyFS with the multi-attribute power of hypersoft sets to solve multi-criteria decision-making problems when faced with extreme uncertainty [23]. Despite these advances, there remains a huge difference between the integration of firm algebraic control and structural constraints into these powerful frameworks [24]. The existing models typically lack mechanisms that can impose consistency and logical composition between sets of parameters, which is fundamental to sound decision-support systems [25]. Our work is inspired by this and suggests the new concepts of PyHSS and Pythagorean super hypersoft sets. Our framework is a combination of the most favorable uncertainty representation of the PyFS, the multi-attribute property of (super)hypersoft sets [26], and the algebraic strength of intersectional conditions [27]. The result of this unification is that all the operations are mathematically closed and logically consistent, unlike previous models. The superiority of our proposed framework has been demonstrated and proven by rigorous mathematical proofs, application of the algorithm on the ground, and application case studies demonstrating that our framework performs better in complex decision-making in digital applications or other high-stakes environments.

1.2 *Research Gap*

The literature review is very exhaustive and provides a clear and critical gap in research. Although current frameworks like PyHSS do offer sophisticated means to address multi-attribute uncertainty, they do not have the algebraic structures required to enable operational closure and logical consistency of complex decision-making processes. Specifically, no single model has been developed that simultaneously satisfies three key properties: the enhanced uncertainty representation of PyFS, which is defined by the property that the total squares of the two parameters, i.e., the squares of the data themselves, add to 1; the multi-attribute parameterization of (super) hypersoft sets, which is necessary to represent complex and interacting factors of the digital supply chain; and the algebraic rigor of intersectional conditions, which is required to inculcate consistency and validity to all set-theoretic operations. This disjunction is particularly pronounced with digital supply chains, where decision-making should be mathematically sound and, at the same time, dynamic enough to address real-time and high-dimensional and often conflicting parameters. Therefore, a new framework that will fill this gap and form a solid base on which supply chain decision-support systems can rely is urgently needed. This paper makes four significant contributions:

- Formal definitions of PySS and PyHSS with complete mathematical characterization.
- The course covers full set-theoretic operations and proofs of closure properties, as well as generalization hierarchies.
- There are numerous digital examples, such as smartphone evaluation, cloud service configuration, and educational recommendation systems.
- We have developed innovative algorithmic and graphical representations of the proposed structures.

The paper is set up like this:Section 2.1 goes over the basic ideas of sets using Algorithm 2. Section 2.2 talks about soft sets and how they are related to the Pythagorean framework. Example 1 in Section 2.3 gives a formal definition of Pythagorean soft sets. Section 2.4 shows basic operations with demonstrations of validity and Table 2. Section 2.5 goes beyond Pythagorean hypersoft sets with Theorem 2.6. Section 2.7 gives a unified view with Algorithm 3. Section 3 shows the major results, which include theoretical properties and advanced uses (Examples 5–6). Lastly, we talk about the findings and what we will do next.

2. Preliminaries

We start by going over some important definitions that are the basis for our new ideas.

Definition 1 *Suppose denote a universe of conversation. A classical set (or crisp set) [28]* C *within can be defined using a characteristic function:*

$$\chi_C :\to \{0,1\}$$

where for any element $x \in$,

$$\chi_C(\breve{x}) = \begin{cases} 1 & \text{if } \breve{x} \in C, \\ 0 & \text{if } \breve{x} \notin C. \end{cases}$$

A classical set is a precisely specified collection of different elements from .

Definition 2 *Suppose denote a universe of conversation. A FSs [17] F in is defined by a MD that assigns each element $\breve{x} \in$ a MD within the interval $[0, 1]$. Formally, F is defined as follows:*

$$F = \{\langle \breve{x}, \psi_F(\breve{x})\rangle \mid \breve{x} \in\}$$

where $\psi_F :\to [0, 1]$ denotes the MD of the element $\breve{x}$ in the FSs F.

Definition 3 *Suppose represent a universe of speech. An IFS [14] I in is an entity characterized by the following structure:*

$$I = \{\langle \breve{x}, \psi_I(\breve{x}), \phi_I(\breve{x})\rangle \mid \breve{x} \in\}$$

where $\psi_I :\to [0, 1]$ and $\phi_I :\to [0, 1]$ delineate the MD and n-MD of the element $\breve{x} \in$ with regard to the set I, respectively. For any $\breve{x} \in$, these functions must adhere to the following condition:

$$0 \leq \psi_I(\breve{x}) + \phi_I(\breve{x}) \leq 1.$$

For any IFS I and each $\breve{x} \in$, the expression $\pi_I(\breve{x}) = 1 - \psi_I(\breve{x}) - \phi_I(\breve{x})$ is referred to as the degree of hesitation or indeterminacy (HD) of $\breve{x}$ with respect to I.

Definition 4 *Suppose denote a universe of conversation. A PyFS [15] P in is an entity characterized by the following structure:*

$$P = \{\langle \breve{x}, \psi_P(\breve{x}), \phi_P(\breve{x})\rangle \mid \breve{x} \in\}$$

where $\psi_P :\to [0, 1]$ and $\phi_P :\to [0, 1]$ delineate the MD and n-MD of the element $\breve{x} \in$ relative to the set P, respectively. For every $\breve{x} \in$, these degrees must adhere to the subsequent relaxed condition:

$$0 \leq (\psi_P(\breve{x}))^2 + (\phi_P(\breve{x}))^2 \leq 1.$$

For each PyFS P and each $\breve{x} \in$, the quantity $\pi_P(\breve{x}) = \sqrt{1 - (\psi_P(\breve{x}))^2 - (\phi_P(\breve{x}))^2}$ is called the HD of $\breve{x}$ to P.

- The defining constraint $\psi^2 + \phi^2 \leq 1$ is less stringent than the IFS constraint $\psi + \phi \leq 1$.
- This permits an expanded range of potential MD/n-MD couples (ψ, ϕ).
- For instance, the pair $(0.7, 0.7)$ is *not* permissible in IFS ($0.7 + 0.7 = 1.4 > 1$) but *is* permissible in PyFS ($0.7^2 + 0.7^2 = 0.49 + 0.49 = 0.98 \leq 1$).
- This attribute enhances PyFSs ability to model intricate and subtle uncertainties.

Algorithm 1 Evolution of Set Theories for Uncertainty Modeling

Input: Universe of discourse , an element $\breve{x} \in$.
Output: The MD characterization of $\breve{x}$ in a set.
procedure Characterize($\breve{x}$)
    Step 1: Classical (Crisp) Set[28]
        $\chi_C(\breve{x}) \leftarrow$ Crisp MD($\breve{x}$) ▷ Returns $\{0, 1\}$
        Property: Absolute belonging. $\chi_C(\breve{x}) = 1$ or 0.

    Step 2: FS [17]
        $\psi_F(\breve{x}) \leftarrow$ Fuzzy MD($\breve{x}$) ▷ Returns $[0, 1]$
        Property: Partial membership. $0 \leq \psi_F(\breve{x}) \leq 1$.

    Step 3: IFS [14]
        $\psi_I(\breve{x}), \phi_I(\breve{x}) \leftarrow$ IFS MD($\breve{x}$)
        Check: $\psi_I(\breve{x}) + \phi_I(\breve{x}) \leq 1$ ▷ Core IFS constraint
        $\pi_I(\breve{x}) \leftarrow 1 - (\psi_I(\breve{x}) + \phi_I(\breve{x}))$ ▷ Hesitancy Degree
        Property: Models MD, n-MD, and HD.

    Step 4: PyFS [15]
        $\psi_P(\breve{x}), \phi_P(\breve{x}) \leftarrow$ PyFS Membership($\breve{x}$)
        Check: $(\psi_P(\breve{x}))^2 + (\phi_P(\breve{x}))^2 \leq 1$ ▷ Core PyFS constraint
        $\pi_P(\breve{x}) \leftarrow \sqrt{1 - (\psi_P(\breve{x}))^2 - (\phi_P(\breve{x}))^2}$ ▷ Pythagorean hesitancy
        Property: Larger domain than IFS. Can model higher uncertainty.

    return (ψ, ϕ, π) ▷ Returns the final tuple based on the chosen model
end procedure

The algorithm (Algorithm 1) and table (Table 1) demonstrate the gradual generalization from classical sets to PyFS. The principal enhancement of PyFS is its less stringent criterion ($\psi^2 + \phi^2 \leq 1$), which accommodates a strictly broader array of MD/n-MD pairs compared to IFS. For example, the pair $(0.7, 0.7)$ is invalid in IFS ($0.7 + 0.7 = 1.4 > 1$) but is valid in PyFS ($0.49 + 0.49 = 0.98 \leq 1$). The enlarged domain, illustrated in the 'Domain' column, enhances PyFS as a more potent and adaptable instrument for modeling intricate, uncertain, and ambiguous data in decision-making processes.

Table 1
Comparison of Set Theories and Their Domains

Set Theory	**Allowed Pair**		**Domain**
	(ψ, ϕ)	(ψ^2, ϕ^2)	
Classical (Crisp) Set	$(1, 0), (0, 1)$	$(1, 0), (0, 1)$	Binary
FS	$[0, 1], 0$	$[0, 1], 0$	MD only
IFS	$\psi + \phi \leq 1$	$\psi + \phi \leq 1$	Smaller
PyFS	$\psi^2 + \phi^2 \leq 1$	$\psi^2 + \phi^2 \leq 1$	Larger

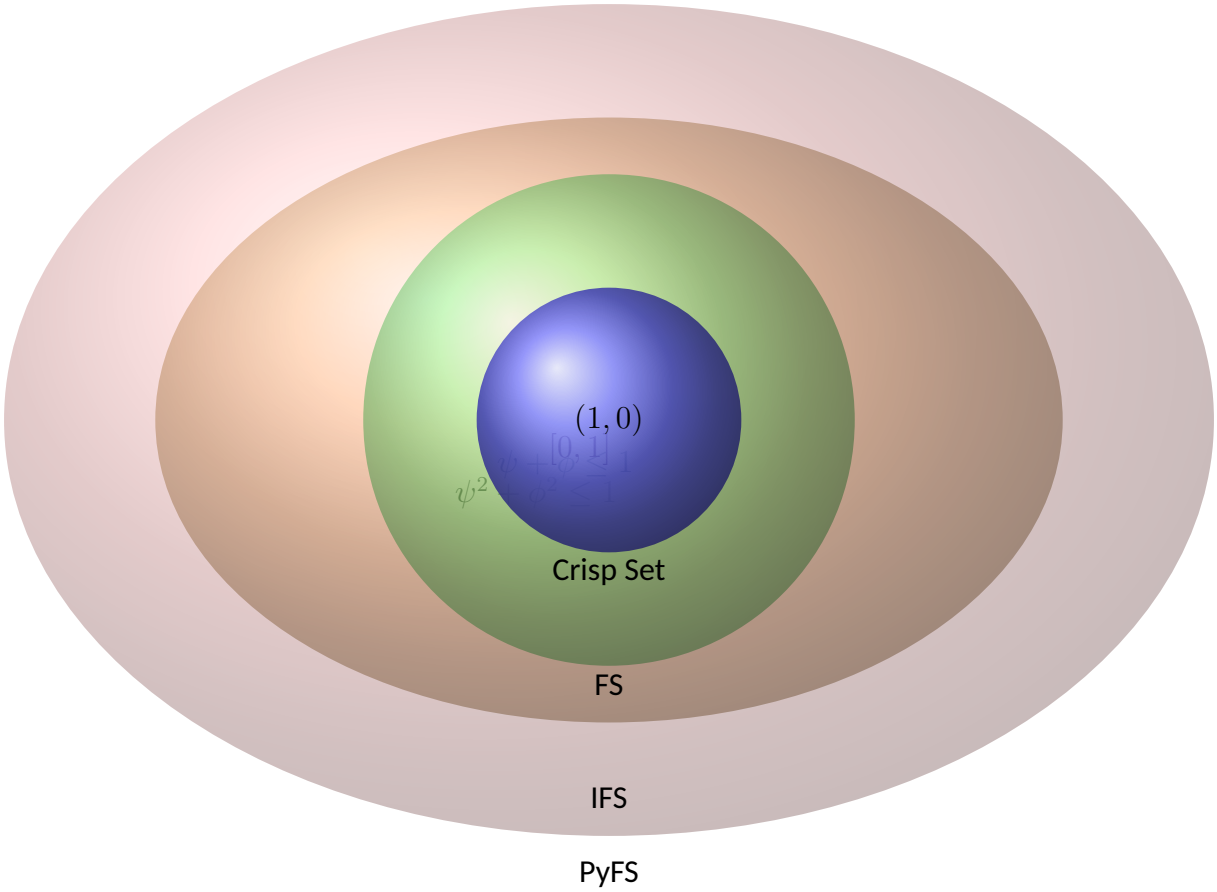


Fig. 1. Illustration of Crisp Set, FS, IFS, and PyFS with Symbols inside each Domain

Figure 1 illustrates the overlapping domains of crisp set, FS, IFS, and PyFS, with each condition contained within its corresponding set.

2.1 Fundamental Set Concepts

We commence by revisiting fundamental set-theoretic notions that form the foundation of soft set theory.

Definition 5 *A well-defined collection of distinct objects is called a set [29]. We write $a \in A$ if a is an element of a set A; if not, we write $a \notin A$.*

Definition 6 *Let's say that A and B are groups. If all the elements in A are also in B, then $A \subseteq B$ is a subset [30] of B. If $A \subseteq B$ but $A \neq B$, then A is a suitable subset of B, which is written as $A \subset B$.*

Definition 7 *The empty set (ES)[31] is the only set that doesn't have any items. It is shown by the symbol $\emptyset$. In formal terms, $\emptyset \subseteq A$ for any set A.*

Definition 8 *Let A denote a set. The collection of all subsets of A is referred to as the power set (PS) [32] of A, denoted by $\mathcal{P}(A)$. Namely,*

$$\mathcal{P}(A) = \{X \mid X \subseteq A\}.$$

Definition 9 *A universal set (US)[33], denoted by , encompasses all elements relevant to a specific context. It is presumed that each set under consideration is a subset of .*

Algorithm 2 delineates the methodology for ascertaining fundamental set properties, encompassing MD, subset relations, the null set, and the formulation of PSs.

Algorithm 2 Determining Fundamental Set Properties and Constructions

procedure Set Properties(A, B,) ▷ Given sets A, B, universal set
    Ensure: $A \subseteq$ and $B \subseteq$

    Step 1: Check Membership
    for each object $\breve{x}$ **under consideration do**
        if $\breve{x}$ is a well-defined object in the context **then**
            $\breve{x}$ belongs to the universe ▷ By Definition 9
        end if
    end for

    Step 2: Check Subset Relation
    $isSubset \leftarrow$ TRUE
    for each element $a \in A$ **do**
        if $a \notin B$ **then**
            $isSubset \leftarrow$ FALSE
            break
        end if
    end for
    Result: $A \subseteq B$ if $isSubset$ is TRUE

    Step 3: Identify Empty Set
    if A has no elements **then**
        $A =$ ▷ By Definition 7
    end if

    Step 4: Construct Power Set
    $\mathcal{P}(A) \leftarrow \{\}$
    for each possible combination X of elements from A **do**
        $\mathcal{P}(A) \leftarrow \mathcal{P}(A) \cup \{X\}$ ▷ By Definition 8
    end for

    return $\mathcal{P}(A)$
end procedure

2.2 Soft Sets

A soft set provides a flexible approach for parameter-driven decision analysis by assigning characteristics (parameters) to subsets of a universal set. This paradigm offers an efficient method for addressing ambiguity and imprecision in complex decision-making processes.

Definition 10 *Let A be a collection of characteristics and be a finite universal set. Let $S \subseteq A$ denote a selected subset of parameters. A soft set (SS)[19] over is a pair (F, S) where*

$$F : S \to \mathcal{P}()$$

is a function that assigns to every parameter $\zeta \in S$ a subset $F(\zeta) \subseteq$. Formally,

$$(F, S) = \{(\zeta, F(\zeta)) \mid \zeta \in S, F(\zeta) \subseteq\}.$$

Connection to Pythagorean Fuzzy Sets: A SS can be thought of as a crisp parameterized model. A natural extension, which we will later call a PySS, would be to swap out the PS $\mathcal{P}()$ for the set of all Pythagorean fuzzy subsets of , $\mathrm{PyFS}()$. This would give us a mapping:

$$F : S \to \mathrm{PyFS}().$$

For for any parameter $\zeta \in S$, $F(\zeta)$ is not a clear subset but a PyFS. P in is made up of all the pairs $\{\langle \breve{x}, \psi_{P(\zeta)}(\breve{x}), \phi_{P(\zeta)}(\breve{x})\rangle \mid \breve{x} \in\}$.

2.3 Pythagorean Fuzzy Soft Sets

Now we will talk about the idea of a PySS.

Definition 11 *Let be a finite universal set and A be a set of parameters. A PySS [6] over , represented as (P, A), is characterized by a mapping:*

$$P : A \to \mathrm{PyFS}(),$$

where $\mathrm{PyFS}()$ is the group of all Pythagorean fuzzy subsets of . For any parameter $\zeta \in A$, $P(\zeta)$ is a PyFS defined as:

$$P(\zeta) = \left\{\langle \breve{x}, \psi_{P(\zeta)}(\breve{x}), \phi_{P(\zeta)}(\breve{x})\rangle \mid \breve{x} \in\right\},$$

with the condition $0 \leq (\psi_{P(\zeta)}(\breve{x}))^2 + (\phi_{P(\zeta)}(\breve{x}))^2 \leq 1$ for all $\breve{x} \in$.

Example 1 *Define the universal set to comprise three smartphone models:*

$$= \{\textit{Model Alpha}, \textit{Model Beta}, \textit{Model Gamma}\}.$$

Designate the parameter set A to denote several features for assessment:

$$A = \{\textit{Camera}, \textit{Battery}, \textit{Design}\}.$$

A PySS (P, A) is established to assess each smartphone model in relation to each characteristic. The mapping $P : A \to \mathrm{PyFS}()$ is defined as follows:

1. *For the parameter $\zeta_1 =$ Camera:*

$$P(\textit{Camera}) = \left\{ \frac{\langle \textit{Model Alpha}, 0.9, 0.3\rangle}{0.9^2 + 0.3^2 = 0.9 \leq 1}, \frac{\langle \textit{Model Beta}, 0.7, 0.5\rangle}{0.7^2 + 0.5^2 = 0.74 \leq 1}, \frac{\langle \textit{Model Gamma}, 0.6, 0.6\rangle}{0.6^2 + 0.6^2 = 0.72 \leq 1} \right\}$$

   (Interpretation: Model Alpha has a high MD (0.9) in good camera quality and a low n-MD (0.3).)

2. *For the parameter $\zeta_2 =$ Battery:*

$$P(\textit{Battery}) = \left\{ \frac{\langle \textit{Model Alpha}, 0.8, 0.4\rangle}{0.8^2 + 0.4^2 = 0.8 \leq 1}, \frac{\langle \textit{Model Beta}, 0.5, 0.7\rangle}{0.5^2 + 0.7^2 = 0.74 \leq 1}, \frac{\langle \textit{Model Gamma}, 0.9, 0.2\rangle}{0.9^2 + 0.2^2 = 0.85 \leq 1} \right\}$$

3. *For the parameter $\zeta_3 =$ Design:*

$$P(\textit{Design}) = \left\{ \frac{\langle \textit{Model Alpha}, 0.7, 0.4\rangle}{0.7^2 + 0.4^2 = 0.65 \leq 1}, \frac{\langle \textit{Model Beta}, 0.9, 0.3\rangle}{0.9^2 + 0.3^2 = 0.9 \leq 1}, \frac{\langle \textit{Model Gamma}, 0.8, 0.3\rangle}{0.8^2 + 0.3^2 = 0.73 \leq 1} \right\}$$

This PySS facilitates a detailed assessment wherein each model's performance is characterized by MD and n-MD (dissatisfaction), consistently adhering to the Pythagorean criterion $\psi^2 + \phi^2 \leq 1$. It accurately encapsulates situations when an element can attain elevated ratings in both MD and n-MD (e.g., Model Beta for Battery: $\langle 0.5, 0.7\rangle$), a phenomenon unattainable in conventional fuzzy or intuitionistic fuzzy SS.

2.4 Fundamental Operations and Their Properties

The efficacy of PySS in decision-making and reasoning is significantly improved by delineating essential set-theoretic procedures. In accordance with the established standards of fuzzy set theory and their extension to the PySS framework [15], we propose the subsequent definitions.

Definition 12 *(P, A) is said to be a pythagorean soft subset of (Q, A), denoted $(P, A)\widetilde{\subseteq}(Q, A)$, if for all $\zeta \in A$ and for all $\breve{x} \in$:*

$$\psi_{P(\zeta)}(\breve{x}) \leq \psi_{Q(\zeta)}(\breve{x}) \quad \textit{and} \quad \phi_{P(\zeta)}(\breve{x}) \geq \phi_{Q(\zeta)}(\breve{x}).$$

Definition 13 *The complement of a PySS (P, A), denoted $(P, A)^c$, is defined for all $\zeta \in A$ and $\breve{x} \in$ by:*

$$P^c(\zeta) = \left\{ \langle \breve{x}, \phi_{P(\zeta)}(\breve{x}), \psi_{P(\zeta)}(\breve{x}) \rangle \mid \breve{x} \in \right\}.$$

Definition 14 *The union of (P, A) and (Q, A), denoted $(P, A)\widetilde{\cup}(Q, A)$, is a PySS (R, A) where for each $\zeta \in A$ and $\breve{x} \in$:*

$$\psi_{R(\zeta)}(\breve{x}) = \max(\psi_{P(\zeta)}(\breve{x}), \psi_{Q(\zeta)}(\breve{x})),$$
$$\phi_{R(\zeta)}(\breve{x}) = \min(\phi_{P(\zeta)}(\breve{x}), \phi_{Q(\zeta)}(\breve{x})).$$

Definition 15 *The intersection of (P, A) and (Q, A), denoted $(P, A)\widetilde{\cap}(Q, A)$, is a PySS (R, A) where for each $\zeta \in A$ and $\breve{x} \in$:*

$$\psi_{R(\zeta)}(\breve{x}) = \min(\psi_{P(\zeta)}(\breve{x}), \psi_{Q(\zeta)}(\breve{x})),$$
$$\phi_{R(\zeta)}(\breve{x}) = \max(\phi_{P(\zeta)}(\breve{x}), \phi_{Q(\zeta)}(\breve{x})).$$

Summary and Validity of Operations

An essential characteristic of these operations is that they are well-defined, i.e., an application of any of them to either a single or two PySSs yields a valid PySS in which the Pythagorean condition holds true of all elements. The operations are summarized below and validated in the table below.

Table 2
Summary of Set-Theoretic Operations for Pythagorean Soft Sets

Operation	Definition	Remarks and Validity
Containment $(P, A)\widetilde{\subseteq}(Q, A)$	$\psi_P \leq \psi_Q$ $\phi_P \geq \phi_Q$	The structure is preserved. The condition $0 \leq \psi_P^2 + \phi_P^2 \leq 1$ for (P, A) ensures $0 \leq \psi_Q^2 + \phi_Q^2 \leq 1$ for (Q, A), as ψ_Q is larger and ϕ_Q is smaller.
Complement $(P, A)^c$	$P^c(\zeta) = \langle \phi_P, \psi_P \rangle$	The Pythagorean condition is invariant: $\phi_P^2 + \psi_P^2 = \psi_P^2 + \phi_P^2 \leq 1$. The operation is an *involution*: $((P, A)^c)^c = (P, A)$.
Union $(P, A)\widetilde{\cup}(Q, A)$	$\psi_R = \max(\psi_P, \psi_Q)$ $\phi_R = \min(\phi_P, \phi_Q)$	Yields a valid PyFS. For any $\breve{x}$, $0 \leq \max(\psi_P, \psi_Q)^2 + \min(\phi_P, \phi_Q)^2 \leq \max(\psi_P^2 + \phi_P^2, \psi_Q^2 + \phi_Q^2) \leq 1$.
Intersection $(P, A)\widetilde{\cap}(Q, A)$	$\psi_R = \min(\psi_P, \psi_Q)$ $\phi_R = \max(\phi_P, \phi_Q)$	Yields a valid PyFS. For any $\breve{x}$, $0 \leq \min(\psi_P, \psi_Q)^2 + \max(\phi_P, \phi_Q)^2 \leq \max(\psi_P^2 + \phi_P^2, \psi_Q^2 + \phi_Q^2) \leq 1$.

Figure 2 depicts the comparative efficacy of containment, complement, union, and intersection operations across the dimensions of consistency, efficiency, closure, and generalization.

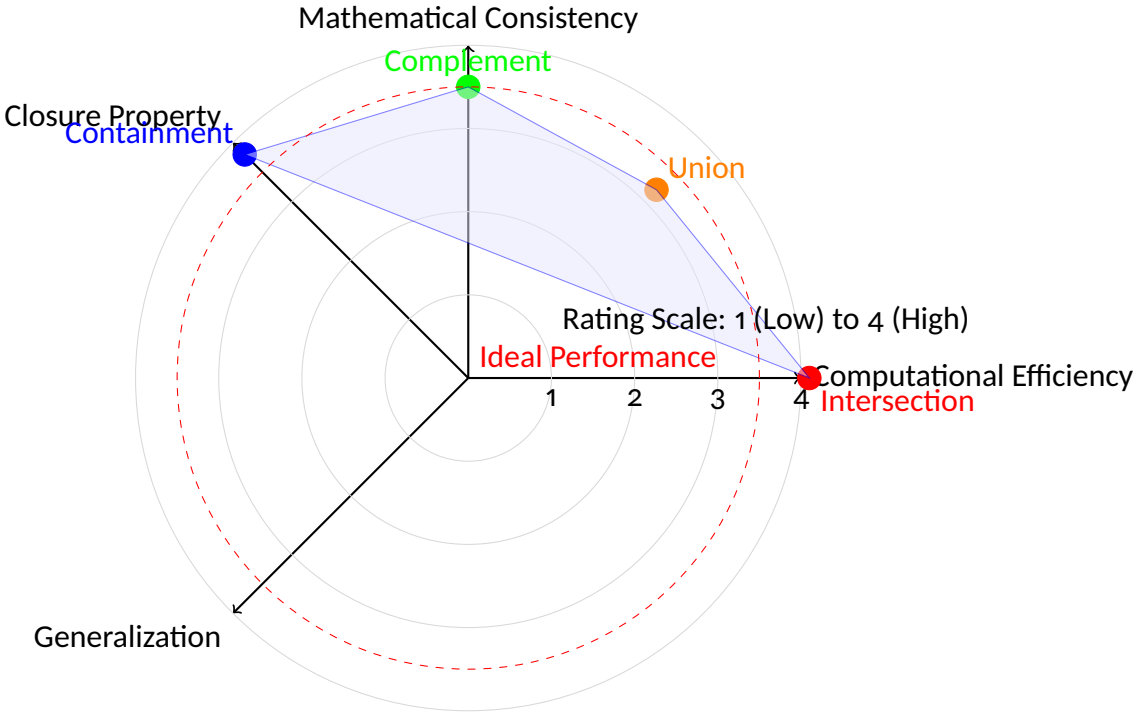


Fig. 2. Operation Performance Evaluation

The observations in Table 2 validate that all operations provide legitimate PyFSs for each parameter $\zeta \in A$. This demonstrates that the collection of all PySSs over a fixed $(, A)$ is closed under these essential set-theoretic operations, thereby creating a resilient algebraic framework for subsequent mathematical investigation and application. The union and intersection operations in PySS are closed. The outcome of the union or intersection of two PySSs is, once more, a PySS. Let $(R, A) = (P, A)\widetilde{\cup}(Q, A)$. We need to show that for all $\zeta \in A$ and $\breve{x} \in$, $0 \leq (\psi_R)^2 + (\phi_R)^2 \leq 1$. Without loss of generality, assume $\psi_R = \psi_P$ and $\phi_R = \phi_Q$. Since (P, A) and (Q, A) are PySS, we have $\psi_P^2 + \phi_P^2 \leq 1$ and $\psi_Q^2 + \phi_Q^2 \leq 1$. It follows that $\psi_P^2 + \phi_Q^2 \leq \psi_P^2 + \phi_P^2 \leq 1$ (if $\phi_Q \leq \phi_P$) or $\psi_P^2 + \phi_Q^2 \leq \psi_Q^2 + \phi_Q^2 \leq 1$ (if $\psi_P \leq \psi_Q$). Thus, the union is a valid PySS. The proof for intersection is analogous.

Example 2 *Suppose* $=$ $\{$*Car1*, *Car2*, *Car3*$\}$ *be a set of cars. Let* $A = \{$*Price*, *Comfort*$\}$ *be the set of parameters. Define a PySS* (P, A) *as follows:*

$$P(\textit{Price}) = \{\langle \textit{Car1}, 0.9, 0.3\rangle, \langle \textit{Car2}, 0.7, 0.5\rangle, \langle \textit{Car3}, 0.5, 0.8\rangle\}.$$
$$P(\textit{Comfort}) = \{\langle \textit{Car1}, 0.6, 0.6\rangle, \langle \textit{Car2}, 0.8, 0.4\rangle, \langle \textit{Car3}, 0.9, 0.2\rangle\}.$$

This PySS models the assessment: for instance, concerning Price, Car1 is deemed highly economical ($\psi = 0.9$) *while not being too costly* ($\phi = 0.3$)*, and observe that* $0.9^2 + 0.3^2 = 0.81 + 0.09 = 0.90 \leq 1$.

2.5 Pythagorean Hypersoft Sets

A hypersoft set enhances multi-attribute decision analysis by associating combinations of many traits with subsets of a universal set, facilitating a more comprehensive and nuanced evaluation.

Definition 16 *Consider m represent the quantity of different attribute domains, and let $A_1, A_2, \ldots, A_m$ denote finite sets. The Cartesian product is the set of all ordered pairs formed by taking one element from each of two sets.*

$$C = A_1 \times A_2 \times \cdots \times A_m,$$

so that each element $\xi \in C$ is an m-tuple

$$\xi = (\xi_1, \xi_2, \ldots, \xi_m) \quad \textit{with} \quad \xi_i \in A_i \textit{ for } i = 1, 2, \ldots, m.$$

A hypersoft set[12] over is a pair (G, C) where

$$G : C \to \mathcal{P}()$$

assigns to each $\xi \in C$ a subset $G(\xi) \subseteq$. Formally,

$$(G, C) = \{(\xi, G(\xi)) \mid \xi \in C, G(\xi) \subseteq\}.$$

Generalization to Pythagorean Framework: A PyHSS extends the concept of a SS by the use of a mapping.

$$G : C \to \mathrm{PFS}().$$

This permits each attribute combination ξ to delineate a PyFS in , facilitating a significantly more expressive framework for multi-dimensional, uncertain decision-making. For each ξ, $G(\xi)$ is a PyFS:

$$G(\xi) = \{\langle \breve{x}, \psi_{G(\xi)}(\breve{x}), \phi_{G(\xi)}(\breve{x})\rangle \mid \breve{x} \in\},$$

with the condition $0 \leq (\psi_{G(\xi)}(\breve{x}))^2 + (\phi_{G(\xi)}(\breve{x}))^2 \leq 1$ for all $\breve{x} \in$. We now extend the PySS to encompass various attribute domains.

Definition 17 *Suppose be a universal set. Let $A_1, A_2, \ldots, A_m$ be m disjoint attribute sets. Let $C = A_1 \times A_2 \times \ldots \times A_m$ be the set of all attribute tuples. A PyHSS [7] over , denoted (P_h, C), is defined by a mapping:*

$$P_h : C \to \mathrm{PyFS}().$$

For each attribute tuple $\xi = (a_1, a_2, \ldots, a_m) \in C$, $P_h(\xi)$ is a PyFS

$$P_h(\xi) = \left\{\langle \breve{x}, \psi_{P_h(\xi)}(\breve{x}), \phi_{P_h(\xi)}(\breve{x})\rangle \mid \breve{x} \in\right\},$$

with $0 \leq (\psi_{P_h(\xi)}(\breve{x}))^2 + (\phi_{P_h(\xi)}(\breve{x}))^2 \leq 1$ for all $\breve{x} \in$.

Example 3 *Suppose the universal set of a corporation represent the "digital base" of electronic gadgets it produces:*

$$= \{\textit{Phone-}X, \textit{Tablet-}Y, \textit{Laptop-}Z\}.$$

The company assesses device compatibility based on two attribute domains:

- $\boldsymbol{A_1}$*:Connection Type =* { *Bluetooth, WiFi* }
- $\boldsymbol{A_2}$*: Security Protocol =* { *WPA2, WPA3* }

The set of all attribute tuples is their Cartesian product:

$$C = A_1 \times A_2 = \{(\textit{Bluetooth, WPA2}), (\textit{Bluetooth, WPA3}), (\textit{WiFi, WPA2}), (\textit{WiFi, WPA3})\}$$

A PyHSS (P_h, C) is constructed to model the degree of compatibility (MD ψ) and degree of vulnerability (n-MD ϕ) for each device under each combined technical attribute. The mapping $P_h : C \to \mathrm{PyFS}()$ is defined as follows:

1. *For $\xi_1 = (\textit{Bluetooth}, \quad \textit{WPA2})$:*

$$P_h(\xi_1) = \left\{ \frac{\langle \textit{Phone-}X, 0.9, 0.3\rangle}{0.9^2 + 0.3^2 = 0.9 \leq 1}, \frac{\langle \textit{Tablet-}Y, 0.7, 0.5\rangle}{0.7^2 + 0.5^2 = 0.74 \leq 1}, \frac{\langle \textit{Laptop-}Z, 0.6, 0.6\rangle}{0.6^2 + 0.6^2 = 0.72 \leq 1} \right\}$$

(Interpretation: Under Bluetooth/WPA2, Phone-X has high compatibility (0.9) and low vulnerability (0.3).)

2. *For $\xi_2 = (\textit{Bluetooth}, \quad \textit{WPA3})$:*

$$P_h(\xi_2) = \left\{ \frac{\langle \textit{Phone-}X, 0.8, 0.4\rangle}{0.8^2 + 0.4^2 = 0.8 \leq 1}, \frac{\langle \textit{Tablet-}Y, 0.5, 0.7\rangle}{0.5^2 + 0.7^2 = 0.74 \leq 1}, \frac{\langle \textit{Laptop-}Z, 0.9, 0.2\rangle}{0.9^2 + 0.2^2 = 0.85 \leq 1} \right\}$$

3. *For $\xi_3 = (\textit{WiFi}, \quad \textit{WPA2})$:*

$$P_h(\xi_3) = \left\{ \frac{\langle \textit{Phone-}X, 0.7, 0.4\rangle}{0.7^2 + 0.4^2 = 0.65 \leq 1}, \frac{\langle \textit{Tablet-}Y, 0.9, 0.3\rangle}{0.9^2 + 0.3^2 = 0.9 \leq 1}, \frac{\langle \textit{Laptop-}Z, 0.8, 0.3\rangle}{0.8^2 + 0.3^2 = 0.73 \leq 1} \right\}$$

4. *For $\xi_4 = (\textit{WiFi}, \quad \textit{WPA3})$:*

$$P_h(\xi_4) = \left\{ \frac{\langle \textit{Phone-}X, 0.6, 0.6\rangle}{0.6^2 + 0.6^2 = 0.72 \leq 1}, \frac{\langle \textit{Tablet-}Y, 0.8, 0.4\rangle}{0.8^2 + 0.4^2 = 0.8 \leq 1}, \frac{\langle \textit{Laptop-}Z, 0.95, 0.1\rangle}{0.95^2 + 0.1^2 = 0.9125 \leq 1} \right\}$$

The PyHSS framework enables a decision-maker to evaluate the performance of each device within the digital infrastructure across all conceivable combinations of connectivity and security parameters, encapsulating the intricate uncertainties (such as high compatibility coupled with high vulnerability, exemplified by $\langle \textit{Tablet-}Y, 0.5, 0.7\rangle$ for ξ_2) that a conventional hypersoft set fails to address. The Pythagorean requirement ($\psi^2 + \phi^2 \leq 1$) is fulfilled for all assessments.

2.6 Operations on PyHSS

The operations of containment, complement, union, and intersection for PyHSS are defined similarly to those for PySS, applied to each attribute tuple $\xi \in C$.

The PyHSS is an extension of both the PySS and the conventional hypersoft set.

1. If $m = 1$ (i.e., there is only one attribute domain), then $C = A_1$ and the PyHSS (P_h, A_1) reduces to a PySS (P, A_1).

2. If the PyFS mappings are restricted to crisp sets (i.e., $\psi(\breve{x}) \in \{0, 1\}, \phi(\breve{x}) = 1 - \psi(\breve{x})$), then the PyHSS (P_h, C) reduces to a standard hypersoft set (G, C).

(i) is evident from the definitions. For (ii), if $\psi_{P_h(\xi)}(\breve{x}) \in \{0, 1\}$ and $\phi_{P_h(\xi)}(\breve{x}) = 1 - \psi_{P_h(\xi)}(\breve{x})$, then the condition $\psi^2 + \phi^2 \leq 1$ still holds ($1^2 + 0^2 = 1, 0^2 + 1^2 = 1$). The mapping $G(\xi) = \{\breve{x} \in | \ \psi_{P_h(\xi)}(\breve{x}) = 1\}$ defines a standard hypersoft set.

Example 4 *A DevOps engineer is considering various options for deploying a model on a cloud platform. The analysis is grounded on multi-dimensional characteristics. Let us define the following:*

- ***Universe ():*** *Set of cloud service configurations.*
  $= \{\textit{Config}_\zeta, \textit{Config}_\lambda, \textit{Config}_\xi\}$.

- ***Attribute Domains:***

$$A_1(\textit{Compute}) = \{\textit{4vCPU}, \textit{8vCPU}\}$$
$$A_2(\textit{Memory}) = \{\textit{16GB}, \textit{32GB}\}$$
$$A_3(\textit{Disk}) = \{\textit{SSD}, \textit{NVMe}\}$$

- ***Set of Attribute Tuples*** (C)***:*** *The Cartesian product of the domains.* $C = A_1 \times A_2 \times A_3$. *For example, one tuple is* $\xi_1 = (\textit{8vCPU}, \textit{32GB}, \textit{NVMe})$.

Cloud performance benchmarks are used to make a PyHSS (P_h, C). *For every attribute tuple (a specific hardware setup), it has a PyFS that rates each software setup in based on performance (MD* ψ*) and cost-inefficiency (n-MD* ϕ*). The PyHSS is a broader version of both the PySS and the normal hypersoft set.*

Attribute Tuple ξ	***Config***	ψ ***(Performance)***	ϕ ***(Cost-Inefficiency)***	***Check*** $\psi^2 + \phi^2 \leq 1$
$\xi_1 =$ *(8vCPU, 32GB, NVMe)*	$\textit{Config}_\zeta$	*0.9*	*0.3*	$0.81 + 0.09 = 0.90$
	$\textit{Config}_\lambda$	*0.7*	*0.6*	$0.49 + 0.36 = 0.85$
	$\textit{Config}_\xi$	*0.5*	*0.8*	$0.25 + 0.64 = 0.89$
$\xi_2 =$ *(4vCPU, 16GB, SSD)*	$\textit{Config}_\zeta$	*0.6*	*0.6*	$0.36 + 0.36 = 0.72$
	$\textit{Config}_\lambda$	*0.8*	*0.4*	$0.64 + 0.16 = 0.80$
	$\textit{Config}_\xi$	*0.9*	*0.2*	$0.81 + 0.04 = 0.85$

Figure 3 shows a radar chart that shows how cloud configurations fit together under the Pythagorean constraint ($\psi^2 + \phi^2 \leq 1$). *The blue and green areas that overlap show where two configurations* (ξ_1 *and* ξ_2) *meet. This shows how performance and cost-inefficiency can be traded off.*

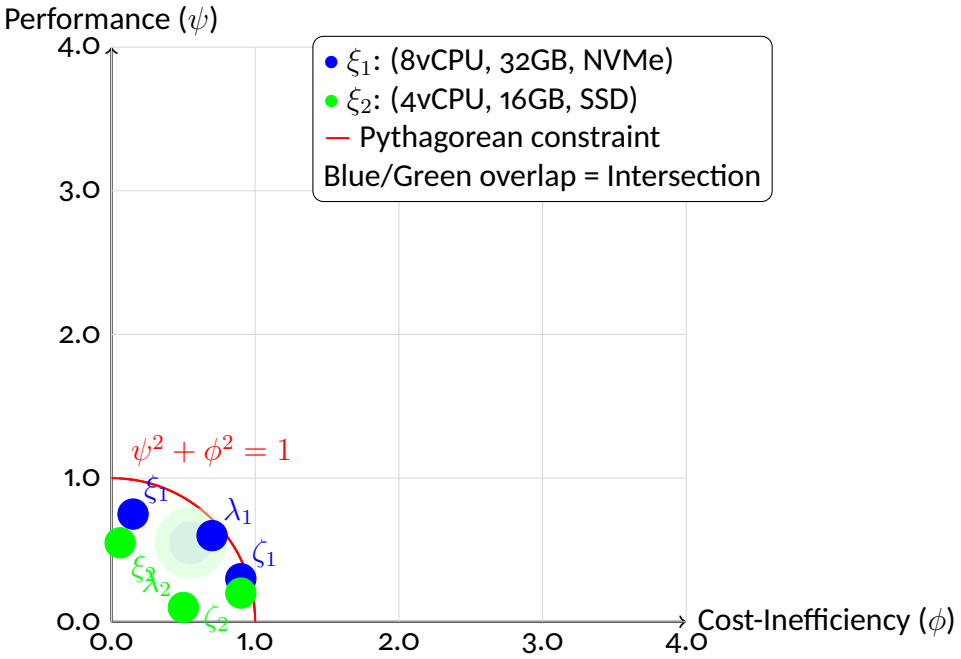


Fig. 3. Visualization of Overlapping Cloud Configurations under Pythagorean Constraint

This PyHSS can help you solve hard questions. For instance, the engineer can restrict the PyHSS to only those tuples using 'NVMe' disks and then execute a union operation over them to identify the optimal configuration for high-performance storage activities. This example also shows the generalization theorem:

1. ***Reduction to PySS:*** *If the engineer just concerned about the "Compute" characteristic ($m = 1$), then $C = A_1 = \{4vCPU, 8vCPU\}$. The PyHSS (P_h, A_1) would then be a PySS that gives each compute option a PyFS and ignores memory and disk.*

2. ***Reduction to Standard Hypersoft Set:*** *If the performance review was just "suitable" or "not suitable" (a clear evaluation), the PyFS mappings would be clear. For instance, for ξ_1 and Config$_\zeta$:*
   If deemed suitable: $P_h(\xi_1)(\text{Config}_\zeta) = \langle 1, 0\rangle$
   If deemed not suitable: $P_h(\xi_1)(\text{Config}_\zeta) = \langle 0, 1\rangle$
   The resulting structure would be a normal hypersoft set (G, C), with $G(\xi)$ being the clear set of configurations that work for that attribute tuple.

This digital example shows how PyHSS gives IT infrastructure a rich, complex framework for making decisions with more than one factor, and how it can generalize simpler models as necessary.

2.7 Unified Framework and Pythagorean Advantages

Algorithm 3 and Table 3 give a complete picture of the PySS and PyHSS frameworks. Their job is to give an organized way to deal with ambiguity in situations when you have to make decisions with one or more attributes. Algorithm 3 is a useful guide for using Pythagorean set structures. It gives a clear plan for how to choose a model (PySS vs. PyHSS), implement it, and test it. The algorithm focuses on the important Pythagorean condition ($\psi^2+\phi^2 \leq 1$), which sets it apart from IFS and lets it model areas with more uncertainty. This enables the depiction of scenarios where $\psi + \phi > 1$ yet $\psi^2 + \phi^2 \leq 1$ (for instance, the pair $(0.7, 0.7)$), a situation that IFS cannot accommodate. Table 3 adds to the algorithm by giving a summary of the important mathematical properties that make the Pythagorean framework strong. The closure attribute for all basic operations (like union and intersection) makes sure that each action on Pythagorean sets (PyS) gives you another valid PyS. The generalization property also reveals that PyHSS includes both PySS and conventional hypersoft sets, which makes it a more flexible and complete model. Compared to IFS, PyS is better because it can represent more things, is mathematically consistent under operations, and can generalize current soft set models. This makes them especially good for scenarios where you have to make a lot of decisions and there are a lot of factors to consider.

Algorithm 3 Overview of Pythagorean Set Structures

procedure ModelUncertainty(, attributes)
    Step 1: Choose Base Structure
    if single attribute domain **then**
        Use PySS framework
        Map: $P : A \to$ PyFS()
    else
        Use PyHSS framework
        Map: $P_h : A_1 \times \ldots \times A_m \to$ PyFS()
    end if

    Step 2: Define Mappings
    for each parameter combination ζ **do**
        for each element $\breve{x} \in$ **do**
            Assign $\psi(\breve{x}) \leftarrow$ MD $\in [0, 1]$
            Assign $\phi(\breve{x}) \leftarrow$ n-MD $\in [0, 1]$
            Ensure: $0 \leq \psi(\breve{x})^2 + \phi(\breve{x})^2 \leq 1$ ▷ Pythagorean condition
        end for
    end for

    Step 3: Perform Operations
    Define containment: $\psi_P \leq \psi_Q$ and $\phi_P \geq \phi_Q$
    Define complement: $P^c(\zeta) = \langle \phi_P, \psi_P \rangle$
    Define union: $\psi_R = \max(\psi_P, \psi_Q)$, $\phi_R = \min(\phi_P, \phi_Q)$
    Define intersection: $\psi_R = \min(\psi_P, \psi_Q)$, $\phi_R = \max(\phi_P, \phi_Q)$

    Step 4: Verify Closure
    for each operation result **do**
        Check $\psi_R^2 + \phi_R^2 \leq 1$ for all $\breve{x} \in$
        Result: All operations yield valid PyFS/PySS/PyHSS
    end for

    Step 5: Apply to Decision Making
    Use for multi-attribute evaluation (Example: smartphone features, cloud configurations)
    Handle uncertainty where $\psi + \phi > 1$ but $\psi^2 + \phi^2 \leq 1$
end procedure

Table 3
Key Properties of Pythagorean Set Structures

Structure	**Key Feature**
PySS	Single attribute domain, Pythagorean fuzzy mappings
PyHSS	Multiple attribute domains, Cartesian product parameter space
Operation	**Definition**
Containment	$\psi_P \leq \psi_Q, \phi_P \geq \phi_Q$
Complement	$P^c(\zeta) = \langle \phi_P, \psi_P \rangle$
Union	$\psi_R = \max(\psi_P, \psi_Q), \phi_R = \min(\phi_P, \phi_Q)$
Intersection	$\psi_R = \min(\psi_P, \psi_Q), \phi_R = \max(\phi_P, \phi_Q)$
Property	**Description**
Closure	All operations produce valid Pythagorean structures
Generalization	PyHSS generalizes both PySS and crisp hypersoft sets
Uncertainty Handling	Supports cases where $\psi + \phi > 1$ but $\psi^2 + \phi^2 \leq 1$

Figure 4 shows how the structures, operations, and properties of PySS and PyHSS work together to allow the integration of Algorithm 3 by making sure that closure, generalization, and strong uncertainty handling are all possible.

3. Main Findings

Figure 4 illustrates the interplay of the structures, operations, and properties of PySS and PyHSS, facilitating the integration of Algorithm 3 by ensuring the feasibility of closure, generalization, and robust uncertainty management.

3.1 Theoretical Properties of Pythagorean Soft Sets

We look into the algebraic structure of PySS based on Definition 13.
Suppose (P, A) be a PySS over . Then the following hold:

1. $(P, A)\widetilde{\cup}(P, A) = (P, A)$
2. $(P, A)\widetilde{\cap}(P, A) = (P, A)$

For any $\zeta \in A$ and $\breve{x} \in$:

$$\psi_{P\widetilde{\cup}P(\zeta)}(\breve{x}) = \max(\psi_{P(\zeta)}(\breve{x}), \psi_{P(\zeta)}(\breve{x})) = \psi_{P(\zeta)}(\breve{x}).$$
$$\phi_{P\widetilde{\cup}P(\zeta)}(\breve{x}) = \min(\phi_{P(\zeta)}(\breve{x}), \phi_{P(\zeta)}(\breve{x})) = \phi_{P(\zeta)}(\breve{x}).$$

Hence, $(P, A)\widetilde{\cup}(P, A) = (P, A)$. The proof for intersection is analogous.
Let (P, A) and (Q, A) be two PySSs over . Then the following identities hold:

1. $((P, A)\widetilde{\cup}(Q, A))^c = (P, A)^c\widetilde{\cap}(Q, A)^c$
2. $((P, A)\widetilde{\cap}(Q, A))^c = (P, A)^c\widetilde{\cup}(Q, A)^c$

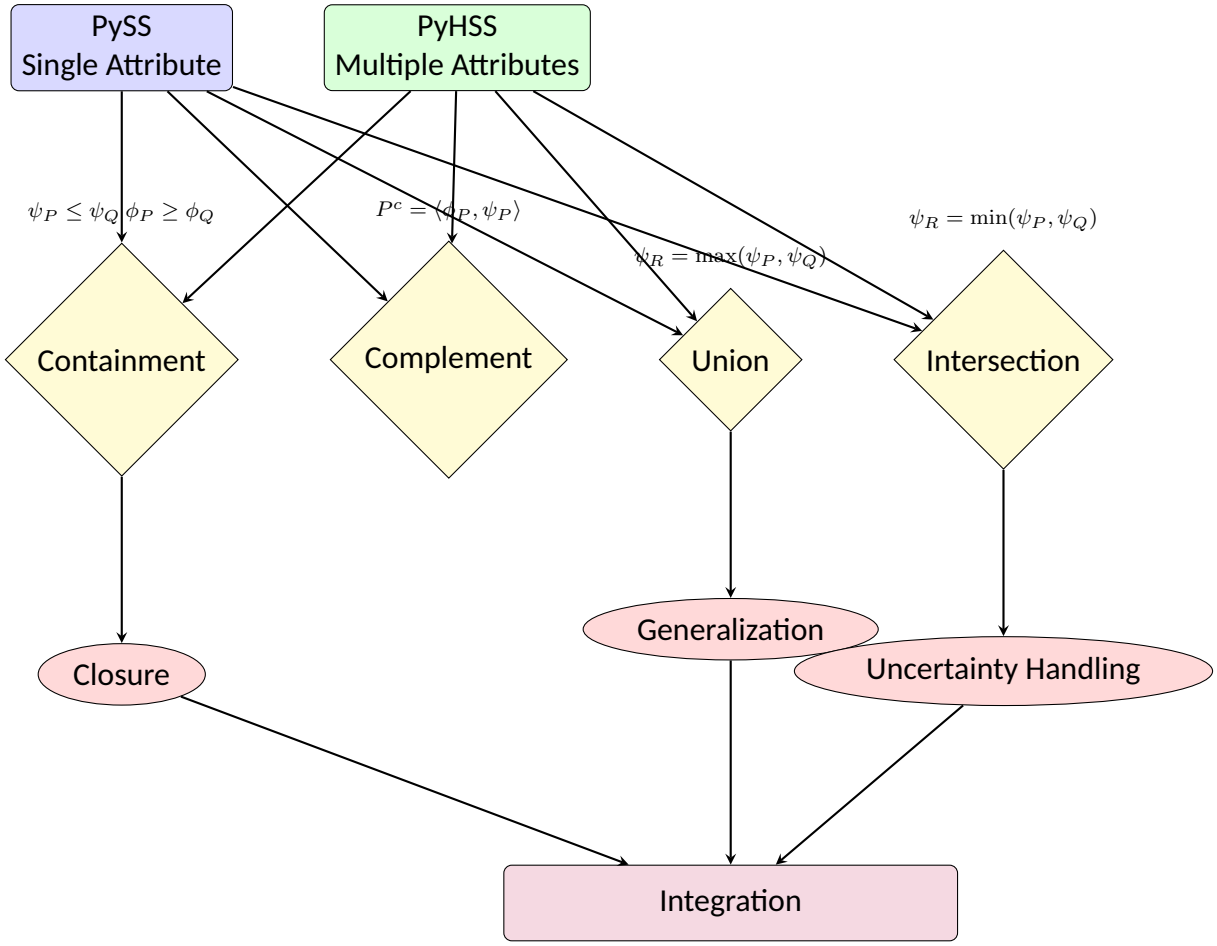


Fig. 4. Representation of Pythagorean Set Structures

We prove the first law. For any $\zeta \in A$ and $\breve{x} \in$:

$$\psi_{((P\tilde{\cup}Q)^c)(\zeta)}(\breve{x}) = \phi_{(P\tilde{\cup}Q)(\zeta)}(\breve{x}) = \min(\phi_{P(\zeta)}(\breve{x}), \phi_{Q(\zeta)}(\breve{x})).$$
$$\text{Also, } \psi_{(P^c\tilde{\cap}Q^c)(\zeta)}(\breve{x}) = \min(\psi_{P^c(\zeta)}(\breve{x}), \psi_{Q^c(\zeta)}(\breve{x})) = \min(\phi_{P(\zeta)}(\breve{x}), \phi_{Q(\zeta)}(\breve{x})).$$

Similarly, for n-MD:

$$\phi_{((P\tilde{\cup}Q)^c)(\zeta)}(\breve{x}) = \psi_{(P\tilde{\cup}Q)(\zeta)}(\breve{x}) = \max(\psi_{P(\zeta)}(\breve{x}), \psi_{Q(\zeta)}(\breve{x})).$$
$$\phi_{(P^c\tilde{\cap}Q^c)(\zeta)}(\breve{x}) = \max(\phi_{P^c(\zeta)}(\breve{x}), \phi_{Q^c(\zeta)}(\breve{x})) = \max(\psi_{P(\zeta)}(\breve{x}), \psi_{Q(\zeta)}(\breve{x})).$$

The identity is still true because the MD and n-MD are the same on both sides. The second law can be demonstrated in a comparable manner.

3.2 An Extended Digital Example: Online Course Selection

Example 5 *Think about an online school where a student can pick a class based on two things: "content quality" and "instructor rating." Think of the universe as a group of courses that are open to everyone:*$= \{$*Course_A*, *Course_B*, *Course_C*$\}$. *Let the collection of parameters be* $A = \{$*Content Quality*, *Instructor*

We construct a PySS (P, A) to represent the platform's AI-driven evaluation:

$$P(\text{Content Quality}) = \{\langle A, 0.9, 0.2\rangle, \langle B, 0.6, 0.5\rangle, \langle C, 0.7, 0.4\rangle\}$$
$$\text{Checks: } 0.9^2 + 0.2^2 = 0.85, 0.6^2 + 0.5^2 = 0.61, 0.7^2 + 0.4^2 = 0.65.$$
$$P(\text{Instructor Rating}) = \{\langle A, 0.8, 0.3\rangle, \langle B, 0.7, 0.6\rangle, \langle C, 0.9, 0.1\rangle\}$$
$$\text{Checks: } 0.8^2 + 0.3^2 = 0.73, 0.7^2 + 0.6^2 = 0.85, 0.9^2 + 0.1^2 = 0.82.$$

Think about an online school where a student can pick a class based on two things: "Content Quality" and "Instructor Rating." Think of the universe as a group of courses that are open to everyone:= {Course_A, Course_B, Course_C}. Let the collection of parameters be $A = \{$Content Quality, Instructor Rating$\}$. We define a PySS (P, A) to represent the platform's AI-driven evaluation:

Course	$\psi_{R(\text{Content})}$	$\phi_{R(\text{Content})}$	Pythagorean Condition
A	$\max(0.9, 0.8) = 0.9$	$\min(0.2, 0.3) = 0.2$	$0.9^2 + 0.2^2 = 0.85 \leq 1$
B	$\max(0.6, 0.7) = 0.7$	$\min(0.5, 0.6) = 0.5$	$0.7^2 + 0.5^2 = 0.74 \leq 1$
C	$\max(0.7, 0.9) = 0.9$	$\min(0.4, 0.1) = 0.1$	$0.9^2 + 0.1^2 = 0.82 \leq 1$

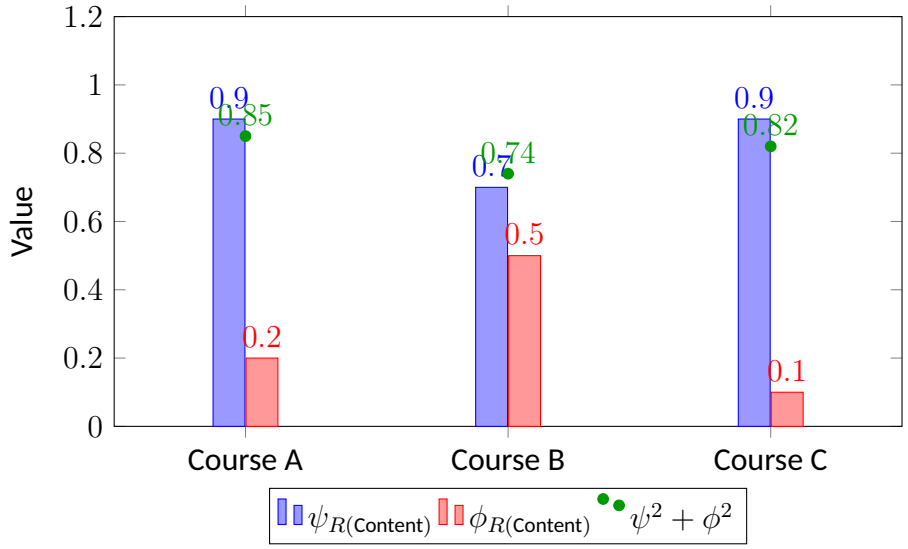


Fig. 5. Statistical Visualization of Course Content under Pythagorean Constraint

This resulting PySS (R, A) enables the learner choose Course_C as a standout alternative ($\psi = 0.9, \phi = 0.1$) depending on any criterion. Figure 5 shows the MD, n-MD, and Pythagorean condition values for different courses on a graph.

3.3 Advanced Operations for Pythagorean Hypersoft Sets

We are now defining more complicated operations for PyHSS that take advantage of its multi-dimensional attribute structure.

Definition 18 *Suppose (P_h, C) be a PyHSS over with $C = A_1 \times A_2 \times \ldots \times A_m$. Let $D \subset C$ be a subset of the attribute tuples. The restriction of (P_h, C) to D is a new PyHSS $(P_h|_D, D)$, defined by:*

$$(P_h|_D)(\xi) = P_h(\xi) \quad \text{for all } \xi \in D.$$

This operation is very important for filtering scenarios based on certain combinations of attributes.

Definition 19 *Suppose (P_h, C) be a PyHSS. Let $w : C \to [0,1]$ be a weighting function such that $\sum_{\xi \in C} w(\xi) = 1$, representing the importance of each attribute combination. The weighted aggregate PyFS $P_{agg} \in \mathrm{PyFS}()$ is defined for each $\breve{x} \in$ as:*

$$\psi_{P_{agg}}(\breve{x}) = \sqrt{\sum_{\xi \in C} w(\xi) \cdot (\psi_{P_h(\xi)}(\breve{x}))^2}$$
$$\phi_{P_{agg}}(\breve{x}) = \sqrt{\sum_{\xi \in C} w(\xi) \cdot (\phi_{P_h(\xi)}(\breve{x}))^2}$$

It can be shown that $0 \leq (\psi_{P_{agg}}(\breve{x}))^2 + (\phi_{P_{agg}}(\breve{x}))^2 \leq 1$ for all $\breve{x} \in$, thus P_{agg} is a valid PyFS.

3.4 Digital Application: Smartphone Configuration Selector

Example 6 *A tech startup sells smart phones that can be changed to fit your needs. A consumer must pick a mix of "RAM" and "Storage." Let's*

$$A_1(RAM) = \{8\text{GB}, 12\text{GB}\}$$
$$A_2(Storage) = \{128\text{GB}, 256\text{GB}, 512\text{GB}\}$$

The universe is the collection of phone models that can be set up: $= \{$Model X, Model Y, Model Z$\}$. The set of attribute tuples is $C = A_1 \times A_2$. The PyHSS (P_h, C) of the company has expert reviews that rate how "suitable" each model is for a certain configuration, taking into account performance (MD) and cost-inefficiency (n-MD). We demonstrate a subset for space:

$$P_h(8\text{GB}, 128\text{GB}) = \{\langle \text{X}, 0.9, 0.2\rangle, \langle \text{Y}, 0.7, 0.5\rangle, \langle \text{Z}, 0.3, 0.8\rangle\}$$
$$P_h(8\text{GB}, 256\text{GB}) = \{\langle \text{X}, 0.8, 0.3\rangle, \langle \text{Y}, 0.8, 0.4\rangle, \langle \text{Z}, 0.5, 0.6\rangle\}$$
$$P_h(12\text{GB}, 512\text{GB}) = \{\langle \text{X}, 0.7, 0.4\rangle, \langle \text{Y}, 0.9, 0.2\rangle, \langle \text{Z}, 0.8, 0.3\rangle\}$$

A customer knows they want at least 256GB storage. We apply an attribute-restriction to $D = \{(8GB, 256GB), (8GB, 512GB), (12GB, 256GB), (12GB, 512GB)\} \subset C$. We combine the limited PyHSS to reach a final recommendation. For all $\xi \in D$, assume that the weight $w(\xi) = 1/4$. We figure up the total score for "Model Y":

$$\psi_{agg}(\text{Y}) = \sqrt{\frac{1}{4}[(0.8)^2 + (0.8)^2 + (0.9)^2]} = \sqrt{\frac{1}{4}[0.64 + 0.64 + 0.81]} = \sqrt{\frac{2.09}{4}} = \sqrt{0.5225} \approx 0.723.$$
$$\phi_{agg}(\text{Y}) = \sqrt{\frac{1}{4}[(0.4)^2 + (0.4)^2 + (0.2)^2]} = \sqrt{\frac{1}{4}[0.16 + 0.16 + 0.04]} = \sqrt{\frac{0.36}{4}} = \sqrt{0.09} = 0.300.$$

Thus, $P_{agg}(\text{Y}) = \langle 0.723, 0.300\rangle$ and $0.723^2 + 0.300^2 \approx 0.523 + 0.090 = 0.613 \leq 1$. Comparing aggregate scores for all models, "Model Y" emerges as a strong, well-balanced choice for the customer's needs.

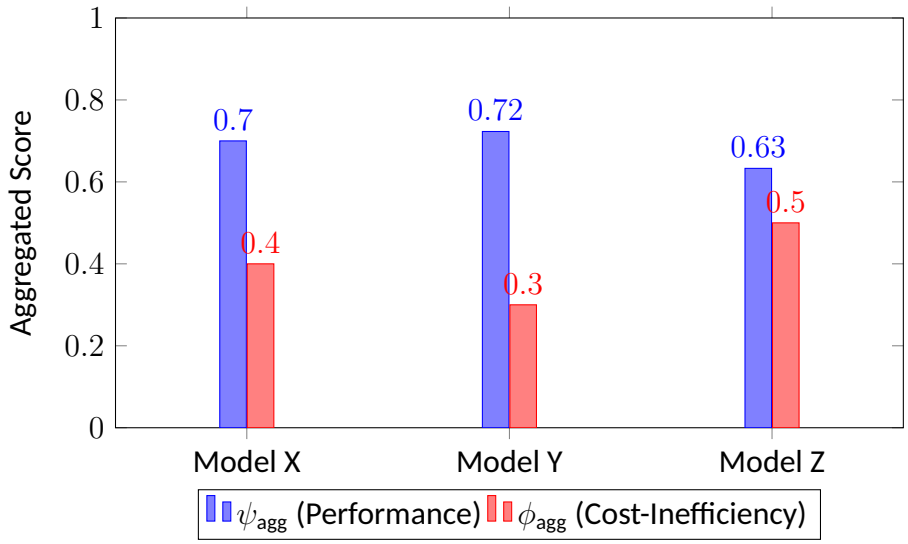


Fig. 6. Aggregated Suitability Scores of Smartphone Models under Attribute Restriction

Figure 6 shows the combined performance and cost-inefficiency scores of the smartphone models. "Model Y" is the best choice because it is the most balanced.

3.5 Theoretical Convergence and Comparison

Every standard hypersoft set (G, C) can be isomorphically embedded into a PyHSS (P_h, C). Suppose (G, C) be a hypersoft set. Define a PyHSS (P_h, C) as follows: for each $\xi \in C$ and $\breve{x} \in$,

$$P_h(\xi)(\breve{x}) = \begin{cases} \langle 1, 0\rangle & \text{if } \breve{x} \in G(\xi), \\ \langle 0, 1\rangle & \text{if } \breve{x} \notin G(\xi). \end{cases}$$

This constitutes a valid PyFS as $1^2 + 0^2 = 1$ and $0^2 + 1^2 = 1$. The mapping $(G, C) \mapsto (P_h, C)$ The function is injective and respects all set-theoretic operations (union, intersection, complement), hence forming an isomorphic embedding. A hierarchy of expressivity exists, wherein each structure is a strict generalization of its predecessor.

Soft SetPySSHypersoft SetPyHSS

The subset relations $\subseteq$ are derived from Theorems 3.1 and 3.1. Strict confinement is illustrated by presenting examples that are elements of one set but not of the other. For example, the PySS in Example 5 cannot be defined as a standard soft set without losing its graded uncertainty information, which shows that SSPySS. In the same way, the PyHSS in Example 6 has information that a PySS with only one attribute domain can't get.

4. Conclusions and Future Work

This research has created a complete mathematical foundation for PySS and PyHSS, showing that they are better than old fuzzy and IFS methods. The results show four main contributions. First, PySS and PyHSS are more expressive because they can show situations where $\psi + \phi > 1$ but $\psi^2 + \phi^2 \leq 1$, which is a major restriction of IFSs. Second, the proposed structures are very mathematically strong since they are closed under basic set-theoretic operations like union, intersection, complement, and containment, as shown in Theorems 2.4, 3.1, and 3.1. Third, their usefulness in real life has been shown

by digital examples in Sections 3.2 and 3.4, which show how well they work for choosing technology, making educational recommendations, and customizing product settings. Finally, PyHSS provides theoretical generality, as demonstrated in Proposition 3.5 and Theorem 3.5, by offering a comprehensive generalization that includes both PySS and standard hypersoft sets, thereby establishing a cohesive structure for MADM under uncertain conditions.

These grounds outline certain promising avenues of study. The first opportunity is to integrate PySS and PyHSS with machine learning, which can enable the development of individual algorithms to classify and predict in a high-uncertainty environment. The other direction of interest is the extension of the theoretical framework to similarity measures, distance measures, and Pythagorean-specific aggregation operators. In terms of their applications, these models have high potential in healthcare decision support, financial risk, and environmental modeling, where uncertainty is inherently complex. Furthermore, cost-efficiency needs to be taken into account in the future work, where the algorithms and data structures must be optimized to efficiently support large-scale PyHSS with a significant amount of combinations of attributes. Finally, this could also be extended theoretically by attempting to find connections with other uncertainty theories such as neutrosophic sets and rough sets, and ultimately by creating hybrid structures to support more holistic decision-making structures.

Acknowledgments

This research was not funded by any grant.

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] AbouElaz, M. A., Alhasnawi, B. N., Sedhom, B. E., & Bureš, V. (2025). Anfis-optimized control for resilient and efficient supply chain performance in smart manufacturing. *Results in Engineering*, *25*, 104262. https://doi.org/10.1016/j.rineng.2025.104262

[2] Kamran, M., Ashraf, S., Abdulla, M. E., & Fatima, S. (2025). Advancing mathematical frontiers: A comprehensive study of the foundations of fermatean fuzzy soft linear spaces and its applications in supply chain management. *Information Sciences*, 122506. https://doi.org/10.1016/j.ins.2025.122506

[3] Yu, X., & Li, N. (2021). Understanding the beginning of a pandemic: China's response to the emergence of covid-19. *Journal of Infection and Public Health*, *14*(3), 347–352. https://doi.org/10.1016/j.jiph.2020.12.024

[4] Kamran, M., Nadeem, M., Żywio lek, J., Abdalla, M. E. M., Uzair, A., & Ishtiaq, A. (2024). Enhancing transportation efficiency with interval-valued fermatean neutrosophic numbers: A multi-item optimization approach. *Symmetry*, *16*(6), 766. https://doi.org/10.3390/sym16060766

[5] Paramesha, M., Rane, N., & Rane, J. (2024). Big data analytics, artificial intelligence, machine learning, internet of things, and blockchain for enhanced business intelligence. *Artificial Intelligence, Machine Learning, Internet of Things, and Blockchain for Enhanced Business Intelligence (June 6, 2024)*. https://doi.org/https://dx.doi.org/10.2139/ssrn.4855856

[6] Kirişci, M., & Şimşek, N. (2022). Decision making method related to pythagorean fuzzy soft sets with infectious diseases application. *Journal of King Saud University-Computer and Information Sciences*, *34*(8), 5968–5978. https://doi.org/10.1016/j.jksuci.2021.08.010

[7] Zulqarnain, R. M., Xin, X. L., & Saeed, M. (2021). A development of pythagorean fuzzy hypersoft set with basic operations and decision-making approach based on the correlation coefficient. *Theory Appl. Hypersoft Set*, *2021*, 85–106. https://doi.org/10.5281/zenodo.4788064

[8] Zheng, L. J., Islam, N., Zhang, J. Z., Behl, A., Wang, X., & Papadopoulos, T. (2025). Aligning risk and value creation: A process model of supply chain risk management in geopolitical disruptions. *International Journal of Operations & Production Management*, *45*(5), 1178–1210. https://doi.org/10.1108/IJOPM-03-2024-0271

[9] Yin, Y., & Yang, Y. (2025). Sustainable transition of the global semiconductor industry: Challenges, strategies, and future directions. *Sustainability*, *17*(7), 3160. https://doi.org/10.3390/su17073160

[10] Kamp, B., Zabala, K., & Zubiaurre, A. (2023). How can machine tool builders capture value from smart services? avoiding the service and digitalization paradox. *Journal of Business & Industrial Marketing*, *38*(2), 303–316. https://doi.org/10.1108/JBIM-12-2021-0588

[11] Deepu, T., & Ravi, V. (2021). Supply chain digitalization: An integrated mcdm approach for inter-organizational information systems selection in an electronic supply chain. *International Journal of Information Management Data Insights*, *1*(2), 100038. https://doi.org/10.1016/j.jjimei.2021.100038

[12] Singh, A. K. (2025). Application of fuzzy logic based multiple-criteria decision making (mcdm) approaches for assessment of sustainability. In *Quantitative assessment of sustainability and sustainable development: Principles, processes, and challenges* (pp. 415–484). Springer. https://doi.org/10.1007/978-3-031-83852-1_12

[13] Rahnamay Bonab, S., & Osgooei, E. (2022). Environment risk assessment of wastewater treatment using fmea method based on pythagorean fuzzy multiple-criteria decision-making. *Environment, Development and Sustainability*, 1–31. https://doi.org/10.1007/s10668-022-02555-5

[14] Atanassov, K. T. (1989). More on intuitionistic fuzzy sets. *Fuzzy sets and systems*, *33*(1), 37–45. https://doi.org/10.1016/0165-0114(89)90215-7

[15] Yager, R. R. (2015). Properties and applications of pythagorean fuzzy sets. In *Imprecision and uncertainty in information representation and processing: New tools based on intuitionistic fuzzy sets and generalized nets* (pp. 119–136). Springer. https://doi.org/10.1016/0165-0114(89)90215-7

[16] Clark, T. D. (2008). *Applying fuzzy mathematics to formal models in comparative politics* (Vol. 225). Springer Science & Business Media. https://doi.org/10.1007/978-3-540-77461-7

[17] Zadeh, L. A. (1965). Fuzzy sets. *Information and control*, *8*(3), 338–353. https://doi.org/10.1016/S0019-9958(65)90241-X

[18] Pan, C., Han, Y., & Lu, J. (2020). Design and optimization of lattice structures: A review. *Applied Sciences*, *10*(18), 6374. https://doi.org/10.3390/app10186374

[19] Sezgin, A., & Atagün, A. O. (2011). On operations of soft sets. *Computers & mathematics with applications*, *61*(5), 1457–1467. https://doi.org/10.1016/j.camwa.2011.01.018

[20] Smarandache, F. (2023). New types of soft sets: Hypersoft set, indetermsoft set, indetermhypersoft set, and treesoft set. *Neutrosophic Systems with Applications*, *8*(1), Article No. 4. https://doi.org/10.61356/j.nswa.2023.41

[21] Al-Odhari, A. (2025). A brief comparative study on hyperstructure, super hyperstructure, and n-super superhyperstructure. *Neutrosophic Knowledge*, *6*, 38–49. https://doi.org/10.5281/zenodo.15107294

[22] Church, A. (1974). Set theory with a universal set. *Proceedings of the Tarski symposium*, *25*, 297–308. https://doi.org/10.1093/oso/9780198514770.001.0001

[23] Khan, K. A., Ishfaq, S. M., Rahman, A. U., & El-Morsy, S. (2024). A synergistic multi-attribute decision-making method for educational institutions evaluation using similarity measures of

possibility pythagorean fuzzy hypersoft sets. *Computer Modeling in Engineering & Sciences*, *142*(1), 501–530. https://doi.org/http://dx.doi.org/10.32604/cmes.2024.057865

[24] Dick, S., Yager, R. R., & Yazdanbakhsh, O. (2015). On pythagorean and complex fuzzy set operations. *IEEE Transactions on Fuzzy Systems*, *24*(5), 1009–1021. https://doi.org/10.1109/TFUZZ.2015.2500273

[25] Wu, W., Ni, Z., Jin, F., Li, Y., & Song, J. (2022). Decision support model with pythagorean fuzzy preference relations and its application in financial early warnings. *Complex & Intelligent Systems*, *8*(1), 443–466. https://doi.org/10.1007/s40747-021-00390-1

[26] Ihsan, M., Saeed, M. H., Alburaikan, A., & Khalifa, H. A. W. (2022). Product evaluation through multi-criteria decision making based on fuzzy parameterized pythagorean fuzzy hypersoft expert set. *AIMS Mathematics*, *7*(6), 11024–11052. https://doi.org/http://dx.doi.org/%2010.3934/math.2022616

[27] Ihsan, M. (2025). Parameterization of multi-decisive hypersoft set under fuzzy environment with application in multi-attribute decision making. *Yugoslav Journal of Operations Research*, (00), 13–13. https://doi.org/10.2298/YJOR241115013I

[28] Komjáth, P., & Totik, V. (2006). *Problems and theorems in classical set theory*. Springer. https://doi.org/10.1007/0-387-36219-3

[29] Van Leeuwen, M., & Knobbe, A. (2012). Diverse subgroup set discovery. *Data Mining and Knowledge Discovery*, *25*(2), 208–242. https://doi.org/10.1007/s10618-012-0273-y

[30] Dong, M., & Kothari, R. (2003). Feature subset selection using a new definition of classifiability. *Pattern Recognition Letters*, *24*(9-10), 1215–1225. https://doi.org/10.1016/S0167-8655(02)00303-3

[31] Sabahi, F., & Akbarzadeh-T, M. R. (2015). Extended fuzzy logic: Sets and systems. *IEEE Transactions on Fuzzy Systems*, *24*(3), 530–543. https://doi.org/10.1109/TFUZZ.2015.2453994

[32] Gitman, V., Hamkins, J. D., & Johnstone, T. A. (2016). What is the theory without power set? *Mathematical Logic Quarterly*, *62*(4-5), 391–406. https://doi.org/10.1002/malq.201500019

[33] Liu, G. (2010). Rough set theory based on two universal sets and its applications. *Knowledge-Based Systems*, *23*(2), 110–115. https://doi.org/10.1016/j.knosys.2009.06.011