CORE CENTER FOR OPEN RESEARCH AND EDUCATION

Applied Research Advances

Journal homepage: www.ara-journal.org
ISSN: 3104-2643

Meta-Graphs and Iterated Meta-Graphs: A Novel Framework for Hierarchical Network Analysis and Complex System Modeling

Takaaki Fujita[1,*], Ajoy Kanti Das[2], Suman Das[3]

[1] Independent Researcher, Tokyo, Japan
[2] Department of Mathematics, Tripura University, Agartala, Tripura, India
[3] Department of Education (ITEP), NIT Agartala, Jirania, Tripura, India

ARTICLE INFO

Article history:
Received 15 February 2026
Received in revised form30 April 2026
Accepted 8 June 2026
Available online 14 June 2026

Keywords:
Meta-Graph, Iterated Meta-Graph, Network Analysis, Hierarchical Modeling, Temporal Networks, Fuzzy Systems

ABSTRACT

Graph theory provides a fundamental framework for representing and analyzing relationships among interconnected entities. In this paper, we introduce the concepts of *Meta-Graphs* and *iterated Meta-Graphs* as hierarchical structures that extend classical graph representations to multiple levels of abstraction. A *Meta-Graph* is defined as a graph whose vertices are themselves graphs, while edges represent specified relations between those graphs. Building on this concept, an *iterated Meta-Graph* is obtained recursively by constructing graphs whose vertices are objects from the preceding level, thereby enabling the representation of complex systems with nested organizational structures. To formalize these ideas, we provide rigorous definitions of Meta-Graphs, relation lifting mechanisms, and iterated universes, together with illustrative examples motivated by real-world networked systems. We further extend the framework by introducing the *Temporal Iterated Meta-Graph*, which models the evolution of hierarchical graph structures through time-indexed snapshots, and the *edge-valued fuzzy iterated Meta-Graph*, which enriches certified meta-relations with membership degrees that quantify uncertainty, confidence, or interaction strength. Several practical examples from microservice architectures, dependency analysis, and organizational systems are presented to demonstrate the applicability of the proposed framework. The introduced concepts establish a foundation for studying higher-order and dynamic graph structures, opening new directions for modeling, analysis, and decision-making in complex interconnected environments.

*Corresponding author.
E-mail address: Takaaki.fujita060@gmail.com

https://doi.org/10.65069/ara31202611

1. Meta-Graphs (Graphs of Graphs)

Graph theory studies mathematical structures composed of vertices and edges to model relationships and connectivity [1]. Informally, a *Meta-Graph* is a graph whose vertices are themselves graphs, and whose edges encode specified relations between those graphs (cf. [2]). Extensions of these concepts are also known in the literature [3–6].

Definition 1.1 (Universe of graphs and relation labels)**.** Let $\mathfrak{G}$ be a nonempty class (or set) of *finite simple undirected graphs* (loopless, without parallel edges) and let $\mathcal{R}$ be a nonempty set of binary relations on $\mathfrak{G}$, i.e.,

$$\mathcal{R} \subseteq \mathcal{P}(\mathfrak{G} \times \mathfrak{G}), \qquad \mathcal{R} \neq \emptyset.$$

Elements $R \in \mathcal{R}$ are called *relation labels*.

Definition 1.2 (Meta-Graph (graph of graphs))**.** (cf. [7]) A *Meta-Graph over* $(\mathfrak{G}, \mathcal{R})$ is a (finite) directed, $\mathcal{R}$-labelled multigraph

$$M = (V, E, s, t, \lambda),$$

where

$$V \subseteq \mathfrak{G}, \qquad E \text{ is a finite set}, \qquad s, t : E \to V, \qquad \lambda : E \to \mathcal{R},$$

such that every meta-edge is *certified* by its label:

$$\forall e \in E : \ (\, s(e),\, t(e)\,) \in \lambda(e).$$

The elements of V are *meta-vertices* (each meta-vertex is a graph $G \in \mathfrak{G}$). For an edge $e \in E$ with $\lambda(e) = R$, we may write $s(e) \xrightarrow{R} t(e)$.

If $\mathcal{R} = \{R\}$ is a singleton, the label map λ may be suppressed. If each $R \in \mathcal{R}$ is symmetric, then the underlying *undirected* labelled multigraph obtained by forgetting orientations is well-defined (equivalently, one may identify opposite orientations).

Example 1.3 (Concrete example of a Meta-Graph)**.** Consider an organization in which each project team has its own *internal interaction graph* whose vertices are employee IDs and whose edges indicate frequent communication. Let $\mathfrak{G}$ be the class of all finite simple undirected graphs whose vertex sets are finite subsets of a fixed label set L.

Define two binary relations on $\mathfrak{G}$:

$$R_{\text{share}} := \big\{(G, H) \in \mathfrak{G} \times \mathfrak{G} \;\big|\; V(G) \cap V(H) \neq \emptyset\big\},$$

$$R_{\text{sub}} := \big\{(H, G) \in \mathfrak{G} \times \mathfrak{G} \;\big|\; H \text{ is isomorphic to a (not necessarily induced) subgraph of } G\big\}.$$

Let $\mathcal{R} := \{R_{\text{share}}, R_{\text{sub}}\}$.

Now define three base graphs (meta-vertices):

$$G_1 : \ a - b - c, \qquad G_2 : \ \text{triangle on } \{c, d, e\}, \qquad G_3 : \ d - e.$$

Then $(G_1, G_2) \in R_{\text{share}}$ since they share the vertex label c, $(G_2, G_3) \in R_{\text{share}}$ since they share d and e, and $(G_3, G_2) \in R_{\text{sub}}$ since the edge $d-e$ occurs as a subgraph of the triangle on $\{c, d, e\}$.

Define the metagraph

$$M = (V, E, s, t, \lambda)$$

by

$$V := \{G_1, G_2, G_3\}, \qquad E := \{e_{12}, e_{23}, e_{32}\},$$

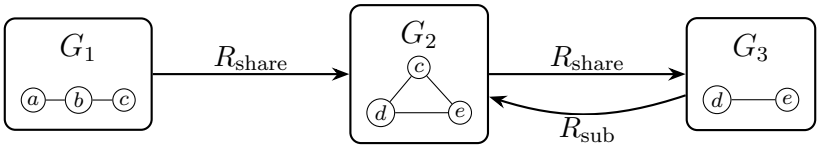

Fig. 1. A Meta-Graph whose vertices are graphs G_1, G_2, G_3. Edges are labelled by relations R_{share} (shared vertex labels) and R_{sub} (subgraph relation).

$$s(e_{12}) = G_1,\ t(e_{12}) = G_2,\ \lambda(e_{12}) = R_{\text{share}},$$
$$s(e_{23}) = G_2,\ t(e_{23}) = G_3,\ \lambda(e_{23}) = R_{\text{share}},$$
$$s(e_{32}) = G_3,\ t(e_{32}) = G_2,\ \lambda(e_{32}) = R_{\text{sub}}.$$

By construction, every meta-edge is certified by its label, so M is a Meta-Graph over $(\mathfrak{G}, \mathcal{R})$ in the sense of Definition 1.2. A schematic drawing is given in Figure 1.

2. Iterated Meta-Graphs (Graphs of Graphs of $\cdots$ of Graphs)

An *iterated Meta-Graph* recursively repeats the construction "graph whose vertices are previous-level objects."

Definition 2.1 (Meta-Graph constructor)**.** For any pair $(\mathfrak{U}, \mathcal{S})$ consisting of a nonempty universe $\mathfrak{U}$ and a nonempty family $\mathcal{S} \subseteq \mathcal{P}(\mathfrak{U} \times \mathfrak{U})$ of binary relations on $\mathfrak{U}$, let

$$\mathrm{Meta}(\mathfrak{U}, \mathcal{S})$$

denote the class of all finite Meta-Graphs over $(\mathfrak{U}, \mathcal{S})$ in the sense of Definition 1.2 (with $\mathfrak{G}, \mathcal{R}$ replaced by $\mathfrak{U}, \mathcal{S}$).

Definition 2.2 (Unit embedding)**.** For $X \in \mathfrak{G}$, define the *unit Meta-Graph* on X by

$$\mathsf{U}(X) := \big(\{X\}, \emptyset, s_\emptyset, t_\emptyset, \lambda_\emptyset\big),$$

where $s_\emptyset, t_\emptyset : \emptyset \to \{X\}$ and $\lambda_\emptyset : \emptyset \to \mathcal{R}$ are the unique functions with empty domain. Then $\mathsf{U} : \mathfrak{G} \hookrightarrow \mathrm{Meta}(\mathfrak{G}, \mathcal{R})$ is injective.

Definition 2.3 (Existential relation lifting)**.** Let $(\mathfrak{U}, \mathcal{S})$ be as in Definition 2.1. For each $S \in \mathcal{S}$ define a binary relation $S^\uparrow$ on $\mathrm{Meta}(\mathfrak{U}, \mathcal{S})$ by

$$(M_1, M_2) \in S^\uparrow \iff \exists\, x \in V(M_1)\, \exists\, y \in V(M_2) :\ (x, y) \in S,$$

where $V(M)$ denotes the vertex set of M. Set

$$\mathcal{S}^\uparrow := \{\, S^\uparrow : S \in \mathcal{S} \,\} \subseteq \mathcal{P}\big(\mathrm{Meta}(\mathfrak{U}, \mathcal{S}) \times \mathrm{Meta}(\mathfrak{U}, \mathcal{S})\big).$$

Definition 2.4 (Iterated universes)**.** Define recursively for $t \in \mathbb{N}_0$:

$$\mathfrak{G}^{(0)} := \mathfrak{G}, \qquad \mathcal{R}^{(0)} := \mathcal{R},$$

and for $t \geq 0$,

$$\mathfrak{G}^{(t+1)} := \mathrm{Meta}\big(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)}\big), \qquad \mathcal{R}^{(t+1)} := \big(\mathcal{R}^{(t)}\big)^\uparrow,$$

where $(\cdot)^\uparrow$ is the lifting from Definition 2.3.

Definition 2.5 (Iterated Meta-Graph of depth t)**.** Let $t \in \mathbb{N}_0$. An *iterated Meta-Graph of depth* t is a Meta-Graph

$$M^{(t)} = \big(V^{(t)}, E^{(t)}, s^{(t)}, t^{(t)}, \lambda^{(t)}\big) \in \mathrm{Meta}\big(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)}\big),$$

that is,

$$V^{(t)} \subseteq \mathfrak{G}^{(t)}, \qquad s^{(t)}, t^{(t)} : E^{(t)} \to V^{(t)}, \qquad \lambda^{(t)} : E^{(t)} \to \mathcal{R}^{(t)},$$

and

$$\forall e \in E^{(t)} : \ \big(s^{(t)}(e),\, t^{(t)}(e)\big) \in \lambda^{(t)}(e).$$

Equivalently, an iterated Meta-Graph of depth t is an element of $\mathfrak{G}^{(t+1)}$.

Remark 2.6. Depth 0 objects are graphs in $\mathfrak{G}$. Depth 1 objects are Meta-Graphs whose vertices are graphs. Depth 2 objects are Meta-Graphs whose vertices are depth 1 Meta-Graphs, and so on. At each depth, every edge is certified by a relation label living at the same depth (obtained via the lifting).

Example 2.7 (Concrete real-world example of an iterated Meta-Graph of depth t)**.** We give a minimal *nontrivial* concrete example by taking depth $t = 1$, i.e. a *graph of Meta-Graphs* (a "graph of graphs of graphs").

Depth 0 (service-internal graphs). Consider an e-commerce platform implemented as microservices. For each microservice S, let $G_S \in \mathfrak{G}$ be the (finite, simple, undirected) *component call graph* of S: vertices are internal modules (billing, fraud-check, cache, ...) and edges indicate frequent runtime calls. Assume we have metadata maps

$$\mathrm{DB}, \mathrm{API} : \mathfrak{G} \to \mathcal{P}(\mathsf{Resources}),$$

where $\mathrm{DB}(G)$ is the set of databases/tables touched by the service represented by G, and $\mathrm{API}(G)$ is the set of external API contracts the service consumes or exposes.

Define two *relation labels* on $\mathfrak{G}$:

$$R_{\mathrm{db}} := \big\{(G,H) \in \mathfrak{G} \times \mathfrak{G} \ \big|\ \mathrm{DB}(G) \cap \mathrm{DB}(H) \neq \emptyset\big\},$$

$$R_{\mathrm{api}} := \big\{(G,H) \in \mathfrak{G} \times \mathfrak{G} \ \big|\ \mathrm{API}(G) \cap \mathrm{API}(H) \neq \emptyset\big\}, \qquad \mathcal{R} := \{R_{\mathrm{db}}, R_{\mathrm{api}}\}.$$

Depth 1 vertices (Meta-Graphs for product domains). Let $\mathfrak{G}^{(1)} = \mathrm{Meta}(\mathfrak{G}, \mathcal{R})$ and $\mathcal{R}^{(1)} = \mathcal{R}^{\uparrow}$ (Definitions 2.1–2.3).

Checkout-domain Meta-Graph. Let $G_{\mathrm{cart}}, G_{\mathrm{pay}}, G_{\mathrm{inv}} \in \mathfrak{G}$ denote the internal graphs of the Cart, Payments, and Inventory services, and suppose (for concreteness)

$$\mathrm{DB}(G_{\mathrm{cart}}) = \{\mathsf{CartDB}\}, \quad \mathrm{DB}(G_{\mathrm{pay}}) = \{\mathsf{OrderDB}, \mathsf{UserDB}\}, \quad \mathrm{DB}(G_{\mathrm{inv}}) = \{\mathsf{StockDB}\},$$

$$\mathrm{API}(G_{\mathrm{cart}}) = \{\mathsf{CheckoutAPI}\}, \quad \mathrm{API}(G_{\mathrm{pay}}) = \{\mathsf{CheckoutAPI}\}, \quad \mathrm{API}(G_{\mathrm{inv}}) = \{\mathsf{ReserveAPI}\}.$$

Define the depth-1 Meta-Graph (a meta-vertex for the next level)

$$M_{\mathrm{chk}} = (V_{\mathrm{chk}}, E_{\mathrm{chk}}, s_{\mathrm{chk}}, t_{\mathrm{chk}}, \lambda_{\mathrm{chk}}) \in \mathrm{Meta}(\mathfrak{G}, \mathcal{R})$$

by

$$V_{\mathrm{chk}} := \{G_{\mathrm{cart}}, G_{\mathrm{pay}}, G_{\mathrm{inv}}\}, \qquad E_{\mathrm{chk}} := \{e_{cp}\},$$

$$s_{\mathrm{chk}}(e_{cp}) = G_{\mathrm{cart}}, \quad t_{\mathrm{chk}}(e_{cp}) = G_{\mathrm{pay}}, \quad \lambda_{\mathrm{chk}}(e_{cp}) = R_{\mathrm{api}}.$$

Indeed, $(G_{\mathrm{cart}}, G_{\mathrm{pay}}) \in R_{\mathrm{api}}$ because both use CheckoutAPI.

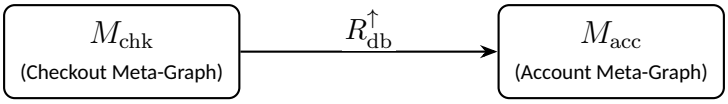

Fig. 2. A depth-1 iterated Meta-Graph $K^{(1)}$ whose vertices are Meta-Graphs $(M_{\mathrm{chk}}, M_{\mathrm{acc}})$. The edge label $R^{\uparrow}_{\mathrm{db}}$ certifies that some underlying pair of services across the two domains shares a database (here via UserDB).

Account-domain Meta-Graph. Let $G_{\mathrm{auth}}, G_{\mathrm{prof}} \in \mathfrak{G}$ denote the internal graphs of the Authentication and Profile services, and suppose

$$\mathrm{DB}(G_{\mathrm{auth}}) = \{\mathsf{UserDB}\}, \quad \mathrm{DB}(G_{\mathrm{prof}}) = \{\mathsf{UserDB}\},$$

$$\mathrm{API}(G_{\mathrm{auth}}) = \{\mathsf{AuthAPI}\}, \quad \mathrm{API}(G_{\mathrm{prof}}) = \{\mathsf{ProfileAPI}\}.$$

Define

$$M_{\mathrm{acc}} = (V_{\mathrm{acc}}, E_{\mathrm{acc}}, s_{\mathrm{acc}}, t_{\mathrm{acc}}, \lambda_{\mathrm{acc}}) \in \mathrm{Meta}(\mathfrak{G}, \mathcal{R})$$

by

$$V_{\mathrm{acc}} := \{G_{\mathrm{auth}}, G_{\mathrm{prof}}\}, \qquad E_{\mathrm{acc}} := \{e_{ap}\},$$
$$s_{\mathrm{acc}}(e_{ap}) = G_{\mathrm{auth}}, \quad t_{\mathrm{acc}}(e_{ap}) = G_{\mathrm{prof}}, \quad \lambda_{\mathrm{acc}}(e_{ap}) = R_{\mathrm{db}}.$$

Indeed, $(G_{\mathrm{auth}}, G_{\mathrm{prof}}) \in R_{\mathrm{db}}$ because both touch UserDB.

Depth 1 Iterated Meta-Graph (graph of Meta-Graphs). Consider the lifted relation $R^{\uparrow}_{\mathrm{db}} \in \mathcal{R}^{(1)}$ on $\mathfrak{G}^{(1)}$:

$$(M_1, M_2) \in R^{\uparrow}_{\mathrm{db}} \iff \exists\, x \in V(M_1)\, \exists\, y \in V(M_2) :\ (x, y) \in R_{\mathrm{db}}.$$

Since $G_{\mathrm{pay}} \in V_{\mathrm{chk}}$ and $G_{\mathrm{auth}} \in V_{\mathrm{acc}}$ satisfy

$$(G_{\mathrm{pay}}, G_{\mathrm{auth}}) \in R_{\mathrm{db}} \quad (\text{because } \mathrm{DB}(G_{\mathrm{pay}}) \cap \mathrm{DB}(G_{\mathrm{auth}}) = \{\mathsf{UserDB}\}),$$

we have $(M_{\mathrm{chk}}, M_{\mathrm{acc}}) \in R^{\uparrow}_{\mathrm{db}}$.

Therefore, the following is a well-defined *iterated Meta-Graph of depth* $t = 1$ (Definition 2.5):

$$K^{(1)} = (V^{(1)}, E^{(1)}, s^{(1)}, t^{(1)}, \lambda^{(1)}) \in \mathrm{Meta}(\mathfrak{G}^{(1)}, \mathcal{R}^{(1)}),$$

where

$$V^{(1)} := \{M_{\mathrm{chk}}, M_{\mathrm{acc}}\}, \qquad E^{(1)} := \{\varepsilon\},$$
$$s^{(1)}(\varepsilon) = M_{\mathrm{chk}}, \quad t^{(1)}(\varepsilon) = M_{\mathrm{acc}}, \quad \lambda^{(1)}(\varepsilon) = R^{\uparrow}_{\mathrm{db}}.$$

The certification constraint holds because $\big(s^{(1)}(\varepsilon), t^{(1)}(\varepsilon)\big) = (M_{\mathrm{chk}}, M_{\mathrm{acc}}) \in R^{\uparrow}_{\mathrm{db}}$.

Interpretation. Vertices M_{chk} and M_{acc} summarize *within-domain* service interactions, while the top-level edge ε records a *cross-domain coupling* induced by shared data resources. Higher depths ($t \geq 2$) arise by grouping these domain Meta-Graphs into larger portfolios (e.g., "Retail", "Identity", "Logistics") and iterating the same construction.

A schematic overview is shown in Figure 2.

3. Temporal Iterated Meta-Graph

A temporal iterated Meta-Graph is a time-indexed family of depth-t Meta-Graphs whose certified labeled meta-edges change across evolving snapshots discretely.

Fix a nonempty universe $\mathfrak{G}$ of finite (simple, undirected) graphs and a nonempty family of binary relations $\mathcal{R} \subseteq \mathcal{P}(\mathfrak{G} \times \mathfrak{G})$. Let $\{\mathfrak{G}^{(t)}, \mathcal{R}^{(t)}\}_{t\in\mathbb{N}_0}$ be the iterated object/relation universes obtained by the Meta-Graph constructor and the chosen lifting (e.g., the existential lifting) as previously defined.

Definition 3.1 (Temporal iterated Meta-Graph of depth t)**.** Let $(\mathbb{T}, \leq)$ be a nonempty totally ordered time set (e.g., $\mathbb{T} = \mathbb{N}$ for discrete time or $\mathbb{T} = \mathbb{R}_{\geq 0}$ for continuous time), and fix $t \in \mathbb{N}_0$.

A *temporal iterated Meta-Graph of depth* t over $(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ is a mapping

$$\mathbb{M}^{(t)} : \mathbb{T} \longrightarrow \mathrm{Meta}\big(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)}\big), \qquad \tau \longmapsto M_\tau^{(t)},$$

where each snapshot

$$M_\tau^{(t)} = \big(V_\tau^{(t)}, E_\tau^{(t)}, s_\tau^{(t)}, t_\tau^{(t)}, \lambda_\tau^{(t)}\big)$$

is a (finite) Meta-Graph over $(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$, i.e.,

$$V_\tau^{(t)} \subseteq \mathfrak{G}^{(t)}, \quad s_\tau^{(t)}, t_\tau^{(t)} : E_\tau^{(t)} \to V_\tau^{(t)}, \quad \lambda_\tau^{(t)} : E_\tau^{(t)} \to \mathcal{R}^{(t)},$$

and satisfies the certification constraint for every $\tau \in \mathbb{T}$:

$$\forall e \in E_\tau^{(t)} : \ \big(s_\tau^{(t)}(e),\, t_\tau^{(t)}(e)\big) \in \lambda_\tau^{(t)}(e).$$

Remark 3.2. One may impose additional regularity conditions depending on the application, e.g. (i) *piecewise-constancy* (finitely many change points on bounded intervals), (ii) vertex-persistence ($V_\tau^{(t)} = V^{(t)}$ for all τ), or (iii) event-time stamping (each edge e carries an activation set $\Theta(e) \subseteq \mathbb{T}$).

Example 3.3 (Concrete real-world example of a temporal iterated Meta-Graph of depth t)**.** We give a concrete *nontrivial* example with depth $t = 1$ and discrete time $\mathbb{T} = \mathbb{N}$, motivated by *incident response and change management* in a microservice system.

Depth 0 **(service-internal graphs).** As in Example 2.7, each microservice S has an internal module-call graph $G_S \in \mathfrak{G}$ (finite, simple, undirected).

Let DB be a fixed set of database resources (tables, topics, buckets, etc.) and assume a metadata map

$$\mathrm{DB} : \mathfrak{G} \to \mathcal{P}(\mathrm{DB}),$$

where $\mathrm{DB}(G)$ is the set of DB resources accessed by the service represented by G.

Define a binary relation label on $\mathfrak{G}$:

$$R_{\mathrm{shareDB}} := \big\{(G, H) \in \mathfrak{G} \times \mathfrak{G} \ \big|\ \mathrm{DB}(G) \cap \mathrm{DB}(H) \neq \emptyset\big\}, \qquad \mathcal{R} := \{R_{\mathrm{shareDB}}\}.$$

Depth 1 **universe and lifted relation.** Let $\mathfrak{G}^{(1)} = \mathrm{Meta}(\mathfrak{G}, \mathcal{R})$ and $\mathcal{R}^{(1)} = \mathcal{R}^{\uparrow}$. Thus $R^{\uparrow}_{\mathrm{shareDB}} \in \mathcal{R}^{(1)}$ is defined by

$$(M_1, M_2) \in R^{\uparrow}_{\mathrm{shareDB}} \iff \exists\, x \in V(M_1)\, \exists\, y \in V(M_2) : \ (x, y) \in R_{\mathrm{shareDB}}.$$

Time axis. Let time be measured in weeks: $\mathbb{T} = \mathbb{N}$, where τ indexes "week τ".

Snapshots (weekly portfolio-of-domains dependency). Fix two domain Meta-Graphs (depth-1 objects)

$$M_{\mathrm{chk}}, M_{\mathrm{acc}} \in \mathfrak{G}^{(1)} = \mathrm{Meta}(\mathfrak{G}, \mathcal{R}),$$

representing the Checkout and Account domains (each is a Meta-Graph whose vertices are service graphs).

For each week $\tau \in \mathbb{N}$, define a depth-1 iterated Meta-Graph snapshot

$$M_\tau^{(1)} = \left(V_\tau^{(1)}, E_\tau^{(1)}, s_\tau^{(1)}, t_\tau^{(1)}, \lambda_\tau^{(1)}\right) \in \mathrm{Meta}(\mathfrak{G}^{(1)}, \mathcal{R}^{(1)})$$

by keeping the *meta-vertex set persistent* and allowing edges to change with deployments:

$$V_\tau^{(1)} := \{M_{\mathrm{chk}}, M_{\mathrm{acc}}\} \quad \text{for all } \tau.$$

Let $E_\tau^{(1)}$ encode whether the two domains share any DB resource *in that week*:

$$E_\tau^{(1)} := \begin{cases} \{\varepsilon\}, & \text{if } (M_{\mathrm{chk}}, M_{\mathrm{acc}}) \in R^{\uparrow}_{\mathrm{shareDB}} \text{ holds in week } \tau, \\ \emptyset, & \text{otherwise.} \end{cases}$$

When $E_\tau^{(1)} = \{\varepsilon\}$, set

$$s_\tau^{(1)}(\varepsilon) = M_{\mathrm{chk}}, \qquad t_\tau^{(1)}(\varepsilon) = M_{\mathrm{acc}}, \qquad \lambda_\tau^{(1)}(\varepsilon) = R^{\uparrow}_{\mathrm{shareDB}}.$$

Concrete change scenario. Suppose that during week $\tau = 10$ a release introduces a new feature where Checkout writes to UserDB (previously used only by Account). Then:

$$E_9^{(1)} = \emptyset \quad \text{(no cross-domain shared DB in week 9)},$$

$$E_{10}^{(1)} = \{\varepsilon\} \quad \text{and} \quad \lambda_{10}^{(1)}(\varepsilon) = R^{\uparrow}_{\mathrm{shareDB}} \quad \text{(shared DB begins in week 10)}.$$

If the feature is later rolled back in week $\tau = 12$, then $E_{12}^{(1)} = \emptyset$ again.

Temporal iterated Meta-Graph. Define

$$\mathbb{M}^{(1)} : \mathbb{N} \to \mathrm{Meta}(\mathfrak{G}^{(1)}, \mathcal{R}^{(1)}), \qquad \tau \longmapsto M_\tau^{(1)}.$$

By construction, every snapshot $M_\tau^{(1)}$ is a Meta-Graph over $(\mathfrak{G}^{(1)}, \mathcal{R}^{(1)})$ and satisfies the certification constraint: if $\varepsilon \in E_\tau^{(1)}$ then

$$\left(s_\tau^{(1)}(\varepsilon), t_\tau^{(1)}(\varepsilon)\right) = (M_{\mathrm{chk}}, M_{\mathrm{acc}}) \in R^{\uparrow}_{\mathrm{shareDB}} = \lambda_\tau^{(1)}(\varepsilon).$$

Hence $\mathbb{M}^{(1)}$ is a *temporal iterated Meta-Graph of depth* $t = 1$ in the sense of Definition 3.1.

Interpretation. The time-indexed snapshots track how cross-domain coupling (here, shared database usage) appears and disappears as the system evolves via deployments and rollbacks, which is central in incident response, auditing, and architectural governance.

Theorem 3.4 (Well-definedness of temporal iterated Meta-Graphs)**.** *Fix $t \in \mathbb{N}_0$ and let $(\mathbb{T}, \leq)$ be a nonempty totally ordered set. Assume that $\mathrm{Meta}(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ denotes the class of all (finite) Meta-Graphs over $(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ as in Definition 1.2. Then Definition 3.1 is well-defined in the following precise sense:*

(i) For each $\tau \in \mathbb{T}$, the snapshot $M_\tau^{(t)} = \big(V_\tau^{(t)}, E_\tau^{(t)}, s_\tau^{(t)}, t_\tau^{(t)}, \lambda_\tau^{(t)}\big)$ is a legitimate Meta-Graph object in $\mathrm{Meta}(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$.

(ii) Consequently, the assignment $\mathbb{M}^{(t)} : \mathbb{T} \to \mathrm{Meta}(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ is a well-typed mapping (i.e., its codomain membership claim is meaningful and satisfied pointwise).

Proof. By Definition 3.1, for each $\tau \in \mathbb{T}$ we are given data

$$V_\tau^{(t)} \subseteq \mathfrak{G}^{(t)}, \qquad E_\tau^{(t)} \text{ finite}, \qquad s_\tau^{(t)}, t_\tau^{(t)} : E_\tau^{(t)} \to V_\tau^{(t)}, \qquad \lambda_\tau^{(t)} : E_\tau^{(t)} \to \mathcal{R}^{(t)},$$

together with the certification constraint

$$\forall e \in E_\tau^{(t)} : \ \big(s_\tau^{(t)}(e),\, t_\tau^{(t)}(e)\big) \in \lambda_\tau^{(t)}(e).$$

This is exactly the defining condition for the tuple $M_\tau^{(t)} = \big(V_\tau^{(t)}, E_\tau^{(t)}, s_\tau^{(t)}, t_\tau^{(t)}, \lambda_\tau^{(t)}\big)$ to be a Meta-Graph over $(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ (cf. Definition 1.2, applied with $\mathfrak{G} \leftarrow \mathfrak{G}^{(t)}$ and $\mathcal{R} \leftarrow \mathcal{R}^{(t)}$). Hence $M_\tau^{(t)} \in \mathrm{Meta}(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ for every $\tau \in \mathbb{T}$, proving (i).

Since the codomain $\mathrm{Meta}(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ is thus satisfied pointwise, the assignment $\tau \mapsto M_\tau^{(t)}$ is a mapping from $\mathbb{T}$ into $\mathrm{Meta}(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ in the set-theoretic sense (i.e., a function with that codomain), proving (ii). □

4. Edge-valued fuzzy iterated Meta-Graph

A fuzzy set assigns each element a membership degree in $[0, 1]$, modeling partial belonging and gradual boundaries under uncertainty contexts. Related concepts are also known, such as Picture Fuzzy Sets[8], Hesitant Fuzzy Sets [9, 10], Neutrosophic Sets [11, 12], and Plithogenic Sets[13–15].

An edge-valued fuzzy iterated Meta-Graph is a depth-t iterated Meta-Graph whose certified labeled meta-edges carry degrees in $[0, 1]$ as weights. Fix $t \in \mathbb{N}_0$ and the depth-t universes $(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ as above.

Definition 4.1 (Edge-valued fuzzy iterated Meta-Graph of depth t)**.** An *edge-valued fuzzy iterated Meta-Graph of depth t* over $(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ is a tuple

$$\widetilde{M}^{(t)} = \big(V^{(t)}, E^{(t)}, s^{(t)}, t^{(t)}, \lambda^{(t)}, \mu^{(t)}\big),$$

where $\big(V^{(t)}, E^{(t)}, s^{(t)}, t^{(t)}, \lambda^{(t)}\big) \in \mathrm{Meta}(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ is an iterated Meta-Graph of depth t and

$$\mu^{(t)} : E^{(t)} \longrightarrow [0, 1]$$

is a *fuzzy edge-strength* (or *edge-membership*) map.

Thus, in addition to

$$V^{(t)} \subseteq \mathfrak{G}^{(t)}, \quad s^{(t)}, t^{(t)} : E^{(t)} \to V^{(t)}, \quad \lambda^{(t)} : E^{(t)} \to \mathcal{R}^{(t)},$$

we require the (crisp) certification constraint

$$\forall e \in E^{(t)} : \ \big(s^{(t)}(e),\, t^{(t)}(e)\big) \in \lambda^{(t)}(e),$$

while $\mu^{(t)}(e)$ quantifies the strength/confidence/intensity of the certified meta-edge e.

Example 4.2 (Concrete real-world example of an edge-valued fuzzy iterated Meta-Graph of depth t)**.** We present a concrete *nontrivial* case with depth $t = 1$ (a fuzzy graph whose vertices are Meta-Graphs). The setting is *enterprise software dependency and risk management* in a microservice architecture.

Depth 0 **(service-internal graphs).** Let each microservice S have an internal (finite, simple, undirected) module-call graph $G_S \in \mathfrak{G}$ as in Example 2.7. Assume a metadata map

$$\mathrm{Deps} : \mathfrak{G} \to \mathcal{P}(\mathsf{Packages}),$$

where $\mathrm{Deps}(G)$ is the set of external libraries/packages (e.g. OSS dependencies) used by the service represented by G.

Define a relation label on $\mathfrak{G}$ capturing *shared dependency*:

$$R_{\mathrm{dep}} := \big\{(G, H) \in \mathfrak{G} \times \mathfrak{G} \;\big|\; \mathrm{Deps}(G) \cap \mathrm{Deps}(H) \neq \emptyset\big\}, \qquad \mathcal{R} := \{R_{\mathrm{dep}}\}.$$

Depth 1 **vertices (Meta-Graphs for product domains).** Let $\mathfrak{G}^{(1)} = \mathrm{Meta}(\mathfrak{G}, \mathcal{R})$ and $\mathcal{R}^{(1)} = \mathcal{R}^{\uparrow}$.

Construct two domain-level Meta-Graphs:

$$M_{\mathrm{chk}} \in \mathrm{Meta}(\mathfrak{G}, \mathcal{R}) \quad \text{(Checkout domain)}, \qquad M_{\mathrm{acc}} \in \mathrm{Meta}(\mathfrak{G}, \mathcal{R}) \quad \text{(Account domain)},$$

where each meta-vertex is a service-internal graph G_S and each meta-edge in the domain is labelled by R_{dep} whenever two services share at least one package.

Depth 1 **crisp certification (lifted relation).** The existential lifting of R_{dep} yields $R_{\mathrm{dep}}^{\uparrow} \in \mathcal{R}^{(1)}$ on $\mathfrak{G}^{(1)}$:

$$(M_1, M_2) \in R_{\mathrm{dep}}^{\uparrow} \iff \exists\, x \in V(M_1)\, \exists\, y \in V(M_2) :\ (x, y) \in R_{\mathrm{dep}}.$$

Assume $(M_{\mathrm{chk}}, M_{\mathrm{acc}}) \in R_{\mathrm{dep}}^{\uparrow}$ holds because there exist services $G_{\mathrm{pay}} \in V(M_{\mathrm{chk}})$ and $G_{\mathrm{auth}} \in V(M_{\mathrm{acc}})$ that share a library, e.g. a TLS or JSON parsing package.

Depth 1 **fuzzy enrichment (edge-valued fuzzy iterated Meta-Graph).** Define the underlying iterated Meta-Graph of depth $t = 1$:

$$M^{(1)} = \big(V^{(1)}, E^{(1)}, s^{(1)}, t^{(1)}, \lambda^{(1)}\big) \in \mathrm{Meta}(\mathfrak{G}^{(1)}, \mathcal{R}^{(1)})$$

by

$$V^{(1)} := \{M_{\mathrm{chk}}, M_{\mathrm{acc}}\}, \qquad E^{(1)} := \{\varepsilon\},$$
$$s^{(1)}(\varepsilon) = M_{\mathrm{chk}}, \quad t^{(1)}(\varepsilon) = M_{\mathrm{acc}}, \quad \lambda^{(1)}(\varepsilon) = R_{\mathrm{dep}}^{\uparrow}.$$

The certification constraint is satisfied because $\big(s^{(1)}(\varepsilon), t^{(1)}(\varepsilon)\big) = (M_{\mathrm{chk}}, M_{\mathrm{acc}}) \in R_{\mathrm{dep}}^{\uparrow}$.

Now attach a fuzzy edge-strength $\mu^{(1)} : E^{(1)} \to [0, 1]$ by quantifying the *strength of the cross-domain dependency coupling*. For instance, let $\mathsf{Crit} : \mathsf{Packages} \to [0, 1]$ be a severity score (e.g. normalized CVSS-like criticality) and define

$$\mu^{(1)}(\varepsilon) := \min\Big(1, \sum_{p \in \mathrm{Deps}(M_{\mathrm{chk}}) \cap \mathrm{Deps}(M_{\mathrm{acc}})} \mathsf{Crit}(p)\Big),$$

where

$$\mathrm{Deps}(M) := \bigcup_{G \in V(M)} \mathrm{Deps}(G)$$

is the set of packages used anywhere in the domain Meta-Graph M.

Finally, define

$$\widetilde{M}^{(1)} = \big(V^{(1)}, E^{(1)}, s^{(1)}, t^{(1)}, \lambda^{(1)}, \mu^{(1)}\big),$$

which is an *edge-valued fuzzy iterated Meta-Graph of depth* $t = 1$ in the sense of Definition 4.1.

Interpretation. The crisp label $R^{\uparrow}_{\mathrm{dep}}$ certifies the *existence* of at least one shared dependency between the Checkout and Account domains. The fuzzy value $\mu^{(1)}(\varepsilon)$ quantifies *how strong or risky* that certified relation is, e.g. higher when many critical packages are shared (or when a small number of shared packages has high criticality). This is useful for prioritizing security remediation and architectural decoupling efforts.

Theorem 4.3 (Well-definedness of edge-valued fuzzy iterated Meta-Graphs)**.** *Fix* $t \in \mathbb{N}_0$. *Assume* $\mathrm{Meta}(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ *is the class of all Meta-Graphs over* $(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$. *Then Definition 4.1 is well-defined in the following sense:*

(i) The tuple $\widetilde{M}^{(t)} = \big(V^{(t)}, E^{(t)}, s^{(t)}, t^{(t)}, \lambda^{(t)}, \mu^{(t)}\big)$ *is a well-typed mathematical object (every component has a specified domain/codomain).*

(ii) The underlying quintuple $\big(V^{(t)}, E^{(t)}, s^{(t)}, t^{(t)}, \lambda^{(t)}\big)$ *is a Meta-Graph in* $\mathrm{Meta}(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$.

(iii) Therefore, $\widetilde{M}^{(t)}$ *defines an* edge-valued fuzzy enrichment *of an iterated Meta-Graph of depth* t *by attaching a membership value in* $[0, 1]$ *to each certified meta-edge.*

Proof. By the hypothesis of Definition 4.1, we start with a quintuple

$$\big(V^{(t)}, E^{(t)}, s^{(t)}, t^{(t)}, \lambda^{(t)}\big) \in \mathrm{Meta}(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)}).$$

Unpacking membership in $\mathrm{Meta}(\mathfrak{G}^{(t)}, \mathcal{R}^{(t)})$ (Definition 1.2), this means precisely that

$$V^{(t)} \subseteq \mathfrak{G}^{(t)}, \qquad s^{(t)}, t^{(t)} : E^{(t)} \to V^{(t)}, \qquad \lambda^{(t)} : E^{(t)} \to \mathcal{R}^{(t)},$$

and the certification constraint holds:

$$\forall e \in E^{(t)} : \ \big(s^{(t)}(e),\, t^{(t)}(e)\big) \in \lambda^{(t)}(e).$$

Next, Definition 4.1 specifies an additional map

$$\mu^{(t)} : E^{(t)} \to [0, 1].$$

Therefore every component of $\widetilde{M}^{(t)} = \big(V^{(t)}, E^{(t)}, s^{(t)}, t^{(t)}, \lambda^{(t)}, \mu^{(t)}\big)$ has a clear domain and codomain, so $\widetilde{M}^{(t)}$ is a well-typed tuple, proving (i).

Moreover, the underlying quintuple is Meta-Graph by assumption (and the above unpacking), proving (ii). Finally, the extra structure $\mu^{(t)}$ assigns to each (already certified) meta-edge $e \in E^{(t)}$ a value $\mu^{(t)}(e) \in [0, 1]$, which is exactly an edge-valued fuzzy enrichment of the iterated Meta-Graph, proving (iii). □

Future research may further extend the proposed framework by incorporating alternative graph-based and uncertainty-aware structures, including neutrosophic graphs [16], plithogenic graphs [17, 18], hypergraphs [19, 20], bidirected graphs [21, 22], and superhypergraphs [23]. Such extensions could provide richer mechanisms for representing uncertainty, higher-order interactions, multidimensional relationships, and complex hierarchical dependencies within Meta-Graph and iterated Meta-Graph environments.

5. Conclusion

In this paper, we introduced the concepts of *Meta-Graphs* and *iterated Meta-Graphs* as hierarchical graph-theoretic structures capable of representing relationships among graphs and among higher-level graph objects. A formal recursive framework was established through the definition of Meta-Graph constructors, relation-lifting mechanisms, and iterated universes, providing a mathematically consistent foundation for multi-level graph representations.

Furthermore, two novel extensions were proposed. The *Temporal Iterated Meta-Graph* enables the modeling of evolving hierarchical graph systems through time-indexed snapshots, while the *edge-valued fuzzy iterated Meta-Graph* incorporates degrees of strength, confidence, or uncertainty into certified meta-relations. These extensions broaden the applicability of the framework to dynamic and uncertain environments.

The presented examples demonstrate that Meta-Graphs and their iterated forms can serve as useful tools for describing complex interconnected systems, including organizational structures, software architectures, dependency networks, and other multi-layered domains. We believe that the proposed framework provides a foundation for future theoretical developments and practical applications involving higher-order relational structures, hierarchical network analysis, and multi-level modeling of complex systems.

Acknowledgement
This research was not funded by any grant.

Conflicts of Interest
The authors declare no conflicts of interest.

References

[1] Diestel, R. (2024). *Graph theory*. Springer (print edition); Reinhard Diestel (eBooks). https://doi.org/10.1007/978-3-662-70107-2

[2] Donnat, C., & Holmes, S. (2018). Tracking network dynamics: A survey using graph distances. *The Annals of Applied Statistics*, *12*(2), 971–1012. https://doi.org/10.1214/18-AOAS1176

[3] Fujita, T., & Smarandache, F. (2026). *Hypergraph and superhypergraph theory with applications*. Neutrosophic Science International Association (NSIA) Publishing House.

[4] Fujita, T. (2025a). Metastructure, meta-hyperstructure, and meta-superhyper structure. *Journal of Computers and Applications*, *1*(1), 1–22. https://doi.org/doi.org/10.58613/jca111

[5] Fujita, T. (2025b). Metahypergraphs, metasuperhypergraphs, and iterated metagraphs: Modeling graphs of graphs, hypergraphs of hypergraphs, superhypergraphs of superhypergraphs, and beyond. *Current Research in Interdisciplinary Studies*, *4*(5), 1–23. https://doi.org/10.58614/cris451

[6] Fujita, T. (2025c). Molecular metagraph and molecular iterated metagraph in chemistryand biochemistry. *Journal of Applied Mathematics and Symbolic Science*, *1*(1), 43–57.

[7] Cao, J., Zhang, S., Chen, Q., Wang, H., Wang, M., & Liu, N. (2022). Network-wide task offloading with leo satellites: A computation and transmission fusion approach. *arXiv preprint arXiv:2211.09672*. https://doi.org/10.48550/arXiv.2211.09672

[8] Cuong, B. C. (2015). Picture fuzzy sets. *Journal of Computer Science and Cybernetics*, *30*, 409. api.semanticscholar.org/CorpusID:55124461

[9] Xu, Z. (2014). *Hesitant fuzzy sets theory* (Vol. 314). Springer. https://doi.org/doi.org/10.1007/978-3-319-04711-9

[10] Torra, V., & Narukawa, Y. (2009). On hesitant fuzzy sets and decision. *2009 IEEE international conference on fuzzy systems*, 1378–1382. https://doi.org/10.1109/FUZZY.2009.5276884

[11] Broumi, S., Smarandache, F., & Dhar, M. (2014). Rough neutrosophic sets. *Infinite Study*, *32*, 493–502. https://doi.org/10.5281/zenodo.30310

[12] Zhao, H., & Zhang, H.-Y. (2020). On hesitant neutrosophic rough set over two universes and its application. *Artificial Intelligence Review*, *53*, 4387–4406. https://doi.org/10.1007/s10462-019-09795-4

[13] Rathinam, N. D., & Parthiban, Y. (2025). A novel approach to sports analytics using neutrosophic fermatean soft plithogenic sets: Application to football team performance evaluation. *Global Integrated Mathematics*, *1*(2), 13–22. https://doi.org/doi.org/10.64229/5dege109

[14] Angel, N., Pandiammal, P., & Martin, N. (2025). Generalized plithogenic cognitive map framework for managerial decision making. *Journal of Applied Research on Industrial Engineering*, *12*(4), 703–713. https://doi.org/10.22105/jarie.2025.530271.1823

[15] Fujita, T., & Smarandache, F. (2025). *A dynamic survey of fuzzy, intuitionistic fuzzy, neutrosophic, and extensional sets*. Neutrosophic Science International Association (NSIA).

[16] Broumi, S., Bakali, A., Talea, M., & Smarandache, F. (2018). An isolated bipolar single-valued neutrosophic graphs. *Information Systems Design and Intelligent Applications: Proceedings of Fourth International Conference INDIA 2017*, 816–822. https://doi.org/10.1007/978-981-10-7512-4_80

[17] Sultana, F., Gulistan, M., Ali, M., Yaqoob, N., Khan, M., Rashid, T., & Ahmed, T. (2023). A study of plithogenic graphs: Applications in spreading coronavirus disease (covid-19) globally. *Journal of ambient intelligence and humanized computing*, *14*(10), 13139–13159. https://doi.org/doi.org/10.1007/s12652-022-03772-6

[18] Campoverde Valencia, E. M., Chuisaca Vásquez, J. P., & Becerra Lois, F. Á. (2025). Multineutrosophic analysis of the relationship between survival and business growth in the manufacturing sector of azuay province, 2020–2023,
using plithogenic n-superhypergraphs. *Neutrosophic Sets and Systems*, *84*(1), 28. https://doi.org/10.5281/zenodo.15640723

[19] Feng, Y., You, H., Zhang, Z., Ji, R., & Gao, Y. (2019). Hypergraph neural networks. *Proceedings of the AAAI conference on artificial intelligence*, *33*(01), 3558–3565. arxiv.org/abs/1809.09401

[20] Cai, D., Song, M., Sun, C., Zhang, B., Hong, S., & Li, H. (2022). Hypergraph structure learning for hypergraph neural networks. *IJCAI*, 1923–1929.

[21] Bang-Jensen, J., & Gutin, G. (2018). *Classes of directed graphs* (Vol. 11). Springer. https://doi.org/10.1007/978-3-319-71840-8

[22] Gellert, L., & Sanyal, R. (2017). On degree sequences of undirected, directed, and bidirected graphs. *European Journal of Combinatorics*, *64*, 113–124. https://doi.org/10.1016/j.ejc.2017.04.002

[23] Smarandache, F. (2020). *Extension of hypergraph to n-superhypergraph and to plithogenic n-superhypergraph, and extension of hyperalgebra to n-ary (classical-/neutro-/anti-) hyperalgebra*. Infinite Study.