Galoitica: Journal of Mathematical Structures and Applications

♦ ISSN: 2834-5568 Vol: 13, No. 2, 2026 ♦ PP. 10–16

Article DOI: https://doi.org/10.54216/GJMSA.130203

FULL LENGTH ARTICLE

On Edge-MetaGraphs

Takaaki Fujita[1,*] **Ajoy Kanti Das**[2] **Suman Das**[3] **Sankar Prasad Mondal**[4] **Volkan Duran**[5]

[1] Independent Researcher, Tokyo, Japan

[2] Associate Professor, Department of Mathematics, Tripura University, Agartala-799022, Tripura, India

[3] Assistant Professor Grade II (Mathematics), Department of Education, National Institute of Technology Calicut, Kozhikode-673601, Kerala, India

[4] Department of Applied Mathematics, Maulana Abul Kalam Azad University of Technology, West Bengal, Haringhata-741249, West Bengal, India

[5] Department of Computer Engineering, Igdır University, Turkey

Emails: Takaaki.fujita060@gmail.com · ajoykantidas@gmail.com · dr.sumandas1995@gmail.com · sankar.mondal02@gmail.com · volkan.duran@igdir.edu.tr

Received: October 30, 2025 Revised: December 26, 2025 Accepted: January 29, 2026 ⋆ Corresponding author

ABSTRACT

This paper studies graph-based higher-order structures related to metagraphs and edge-labeled hierarchical networks. After reviewing MetaGraphs and Iterated MetaGraphs, we introduce the notion of an Edge-MetaGraph, in which each edge is labeled by a two-ported internal graph, allowing edge-substitution expansion through port gluing. We then define Iterated Edge-MetaGraphs recursively, so that edges may carry nested Edge-MetaGraph structures. Concrete examples from biomedical systems, software pipelines, and logistics are presented to illustrate the expressive power of the proposed framework.

Keywords: Edge-MetaGraph ▪ Iterated Edge-MetaGraph ▪ MetaGraph ▪ Iterated MetaGraph

1. PRELIMINARIES

This section establishes notation and summarizes the set-theoretic and combinatorial notions needed in the sequel. Throughout, all sets that serve as vertex domains are assumed to be finite, and we write $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$.

1.1 MetaGraph (Graph of Graphs)

Graph theory investigates mathematical structures consisting of vertices and edges for modeling relationships and connectivity [1]. However, ordinary graphs may not always provide sufficient flexibility for certain applications. To address such limitations, various extensions of graph theory have been developed, including *Fuzzy Graphs* [2, 3] and *Neutrosophic Graphs* [4, 5, 6]. Another notable extension is the *MetaGraph*. A *MetaGraph* is a graph whose vertices are themselves graphs, with edges representing specified relations between those graphs (cf. [7, 8, 9]). A MetaGraph is also known as a *Graph of Graphs* (cf. [10, 11, 12]). Related generalized graph structures capable of representing higher-order or structurally complex relationships, such as *HyperGraphs*[13, 14, 15] and *SuperHyperGraphs*[16, 17], are also well known.

Definition 1.1 (Metagraph (graph of graphs))**.** [18] Fix a nonempty universe $\mathfrak{G}$ of finite graphs (undirected, loopless by default) and a nonempty family of binary relations

$$\mathscr{R} \subseteq \mathscr{P}\big(\mathfrak{G} \times \mathfrak{G}\big).$$

A *metagraph over* $(\mathfrak{G}, \mathscr{R})$ is a directed, labelled multigraph

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

with

$$V \subseteq \mathfrak{G}, \qquad s, t : E \to V, \qquad \lambda : E \to \mathscr{R},$$

satisfying the incidence constraint

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

Elements of V are *meta-vertices* (each is a graph $G \in \mathfrak{G}$). For $e \in E$ with $\lambda(e) = R$, we write $s(e) \xrightarrow{R} t(e)$ and call e a *meta-edge*. If $\mathscr{R} = \{R\}$ is a singleton, labels may be omitted. If every $R \in \mathscr{R}$ is symmetric, M can be viewed as an undirected labelled multigraph.

Remark 1.2. We fix a common base universe $\mathfrak{G}$ of finite, loopless graphs that encode biochemical modules. Each $G \in \mathfrak{G}$ comes with a vertex-labeling $\ell_V : V(G) \to \Sigma_V$ (e.g. metabolite/protein/gene names) and, when needed, an edge-labeling $\ell_E : E(G) \to \Sigma_E$ (e.g. reaction/interaction types). We use the following chemically meaningful binary relations on $\mathfrak{G}$:

$$R_{\text{shareM}}(G,H) : \Longleftrightarrow \{\text{metabolite names in } G\} \cap \{\text{metabolite names in } H\} \neq \emptyset,$$

$$R_{\text{shareP}}(G,H) : \Longleftrightarrow \{\text{protein names in } G\} \cap \{\text{protein names in } H\} \neq \emptyset,$$

$$R_{\text{GRN}\Delta}(G,H) : \Longleftrightarrow E(G) \triangle E(H) \neq \emptyset,$$

for directed GRNs with activation or inhibition labels.

Set $\mathscr{R} := \{R_{\text{shareM}}, R_{\text{shareP}}, R_{\text{GRN}\Delta}\}$.

Example 1.3 (A metagraph based on shared vertices)**.** Let

$$\mathfrak{G} = \{G_1, G_2, G_3\},$$

where the three finite undirected graphs are defined by

$$G_1 = (V_1, E_1), \qquad G_2 = (V_2, E_2), \qquad G_3 = (V_3, E_3),$$

with

$$V_1 = \{a,b,c\}, \qquad E_1 = \big\{\{a,b\},\{b,c\}\big\},$$

$$V_2 = \{c,d,e\}, \qquad E_2 = \big\{\{c,d\},\{d,e\}\big\},$$

and

$$V_3 = \{e,f,g\}, \qquad E_3 = \big\{\{e,f\},\{f,g\}\big\}.$$

Define a binary relation

$$R_{\text{share}} \subseteq \mathfrak{G} \times \mathfrak{G}$$

by

$$(G_i, G_j) \in R_{\text{share}} \quad \Longleftrightarrow \quad V(G_i) \cap V(G_j) \neq \varnothing.$$

Thus, two graphs are related whenever they share at least one vertex. Let

$$\mathscr{R} = \{R_{\text{share}}\}.$$

In the present example,

$$V(G_1) \cap V(G_2) = \{c\},$$

and hence

$$(G_1, G_2) \in R_{\text{share}}.$$

Similarly,

$$V(G_2) \cap V(G_3) = \{e\},$$

so that

$$(G_2, G_3) \in R_{\text{share}}.$$

On the other hand,

$$V(G_1) \cap V(G_3) = \varnothing,$$

and therefore

$$(G_1, G_3) \notin R_{\text{share}}.$$

We may consequently define the metagraph

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

by

$$V = \{G_1, G_2, G_3\}, \qquad E = \{\varepsilon_{12}, \varepsilon_{23}\},$$

with

$$s(\varepsilon_{12}) = G_1, \qquad t(\varepsilon_{12}) = G_2, \qquad \lambda(\varepsilon_{12}) = R_{\text{share}},$$

and

$$s(\varepsilon_{23}) = G_2, \qquad t(\varepsilon_{23}) = G_3, \qquad \lambda(\varepsilon_{23}) = R_{\text{share}}.$$

The incidence condition is satisfied because

$$\big(s(\varepsilon_{12}), t(\varepsilon_{12})\big) = (G_1, G_2) \in R_{\text{share}}$$

and

$$\big(s(\varepsilon_{23}), t(\varepsilon_{23})\big) = (G_2, G_3) \in R_{\text{share}}.$$

Hence the resulting metagraph may be written as

$$G_1 \xrightarrow{R_{\text{share}}} G_2 \xrightarrow{R_{\text{share}}} G_3.$$

Notice that the vertices of M are not ordinary vertices: each meta-vertex G_i is itself an entire graph. The relation labels on the meta-edges specify why the corresponding graphs are connected at the metagraph level.

Figure 1 illustrates this construction.

1.2 Iterated MetaGraph (Graph of Graphs of ... of Graphs)

An Iterated MetaGraph is a graph whose vertices are metagraphs, recursively extending graph-of-graphs structure to multiple hierarchical levels [18]. Related concepts, such as MetaStructures[18], are also known.

Definition 1.4 (Unit metagraph embedding)**.** For $X \in \mathfrak{G}$ define the *unit metagraph*

$$\mathsf{U}(X) \ := \ \big(\{X\}, \emptyset, \_, \_, \_\big).$$

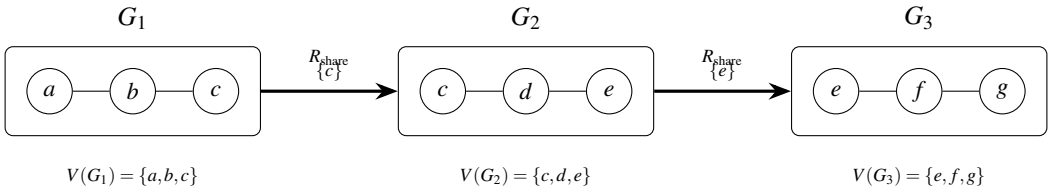
This gives an injective map $\mathsf{U} : \mathfrak{G} \hookrightarrow \mathrm{Obj}\big(\mathrm{Meta}(\mathfrak{G}, \mathscr{R})\big)$.

Definition 1.5 (Relation lifting)**.** For each $R \in \mathscr{R}$, define its lift $R^{\uparrow}$ on finite metagraphs over $(\mathscr{G}, \mathscr{R})$ by

$$(M_1, M_2) \in R^{\uparrow} \Longleftrightarrow \ \exists x \in V(M_1), \ \exists y \in V(M_2) \text{ such that } (x,y) \in R.$$

Set

$$\mathscr{R}^{\uparrow} := \{R^{\uparrow} : R \in \mathscr{R}\}.$$



Metagraph level: $G_1 \xrightarrow{R_{\text{share}}} G_2 \xrightarrow{R_{\text{share}}} G_3$

Figure 1. A metagraph whose meta-vertices are the graphs G_1, G_2, and G_3. The meta-edge $G_1 \to G_2$ exists because the two graphs share the vertex c, while $G_2 \to G_3$ exists because they share the vertex e. Since $V(G_1) \cap V(G_3) = \varnothing$, there is no direct meta-edge between G_1 and G_3.

Definition 1.6 (Iterated object and relation universes)**.** Define recursively for $t \in \mathbb{N}_0$:

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

$$\mathfrak{G}^{(t+1)} := \left\{ \begin{array}{c} \text{finite metagraphs over} \\ \left(\mathfrak{G}^{(t)}, \mathscr{R}^{(t)}\right) \end{array} \right\},$$

$$\mathscr{R}^{(t+1)} := \left(\mathscr{R}^{(t)}\right)^{\uparrow}.$$

Definition 1.7 (Iterated MetaGraph of depth t)**.** For $t \in \mathbb{N}_0$, an *iterated metagraph of depth t* is a metagraph

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

over $\left(\mathfrak{G}^{(t)}, \mathscr{R}^{(t)}\right)$, i.e., $V^{(t)} \subseteq \mathfrak{G}^{(t)}$, $\lambda^{(t)} : E^{(t)} \to \mathscr{R}^{(t)}$ and

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

Remark 1.8. We use the relation lifting of the preliminaries: for $R \in \mathscr{R}$ and metagraphs $\mathsf{M}_1, \mathsf{M}_2$ over $(\mathfrak{G}, \mathscr{R})$,

$$(\mathsf{M}_1, \mathsf{M}_2) \in R^{\uparrow} \iff \begin{array}{l} \exists X \in V(\mathsf{M}_1), \\ \exists Y \in V(\mathsf{M}_2) \text{ such that } (X,Y) \in R. \end{array}$$

Example 1.9 (A concrete iterated metagraph (depth $t = 1$) with a biomedical interpretation)**.** We give an explicit example of an iterated metagraph of depth $t = 1$ (i.e., a graph whose vertices are metagraphs of base graphs).

Base level ($\mathfrak{G}^{(0)} = \mathfrak{G}$). Let $\mathfrak{G}$ contain three pathway graphs

$$\begin{aligned} G_1 &= \mathsf{Glycolysis}, \\ G_2 &= \mathsf{Pyruvate\ Oxidation}, \\ G_3 &= \mathsf{TCA\ Cycle}. \end{aligned}$$

Let $\mathscr{R}^{(0)} = \{R_{\text{shareM}}\}$, where

$$(G,H) \in R_{\text{shareM}} \iff \begin{array}{l} \text{the pathways } G \text{ and } H \\ \text{share at least one metabolite.} \end{array}$$

Assume $(G_1, G_2) \in R_{\text{shareM}}$ (shared metabolite: pyruvate) and $(G_2, G_3) \in R_{\text{shareM}}$ (shared metabolite: acetyl-CoA).

Metagraphs of graphs (objects in $\mathfrak{G}^{(1)}$). Define two metagraphs over $(\mathfrak{G}^{(0)}, \mathscr{R}^{(0)})$:

$$M_A^{(0)} : \ V(M_A^{(0)}) = \{G_1, G_2\}, \text{ one meta-edge } G_1 \xrightarrow{R_{\text{shareM}}} G_2,$$

$$M_B^{(0)} : \ V(M_B^{(0)}) = \{G_2, G_3\}, \text{ one meta-edge } G_2 \xrightarrow{R_{\text{shareM}}} G_3.$$

By construction, $M_A^{(0)}, M_B^{(0)} \in \mathfrak{G}^{(1)}$.

Lifted relation and depth-1 iterated metagraph. Since $(G_2, G_3) \in R_{\text{shareM}}$ with $G_2 \in V(M_A^{(0)})$ and $G_3 \in V(M_B^{(0)})$, the lifted relation satisfies

$$\left(M_A^{(0)}, M_B^{(0)}\right) \in R_{\text{shareM}}^{\uparrow} \in \mathscr{R}^{(1)}.$$

Hence the depth-1 iterated metagraph

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

can be defined by

$$\begin{aligned} V^{(1)} &= \{M_A^{(0)}, M_B^{(0)}\}, & E^{(1)} &= \{e\}, \\ s^{(1)}(e) &= M_A^{(0)}, & t^{(1)}(e) &= M_B^{(0)}, \\ \lambda^{(1)}(e) &= R_{\text{shareM}}^{\uparrow}. \end{aligned}$$

Figure 2 depicts this construction.

2. RESULTS

The results of this paper are presented in this section.

2.1 Edge-MetaGraph (Graph-labeled edges)

An Edge-MetaGraph is a directed multigraph whose edges carry two-ported internal graphs, enabling edge-substitution expansion by gluing ports to endpoints.

Definition 2.1 (Two-ported graphs)**.** Fix a nonempty universe $\mathfrak{G}$ of finite (undirected, loopless by default) graphs. A *two-ported graph* is a triple

$$H^{\partial} = (H, p^-, p^+),$$

where $H \in \mathfrak{G}$ and $p^-, p^+ \in V(H)$ are distinguished vertices called the *tail-port* and *head-port*, respectively. We write $\mathrm{Tail}(H^{\partial}) := p^-$ and $\mathrm{Head}(H^{\partial}) := p^+$, and denote by

$$\mathfrak{G}^{\partial 2} := \{(H, p^-, p^+) : H \in \mathfrak{G}, \ p^-, p^+ \in V(H)\}$$

the class of all two-ported graphs over $\mathfrak{G}$.

Definition 2.2 (Edge-MetaGraph)**.** Let $\mathfrak{G}^{\partial 2}$ be as in Definition 2.1. An *Edge-MetaGraph over* $\mathfrak{G}$ is a directed multigraph equipped with a two-ported-graph label on each edge, i.e., a tuple

$$\mathbb{E} \ = \ (V, E, s, t, \ell),$$

where (V, E, s, t) is a directed multigraph (loops and parallel

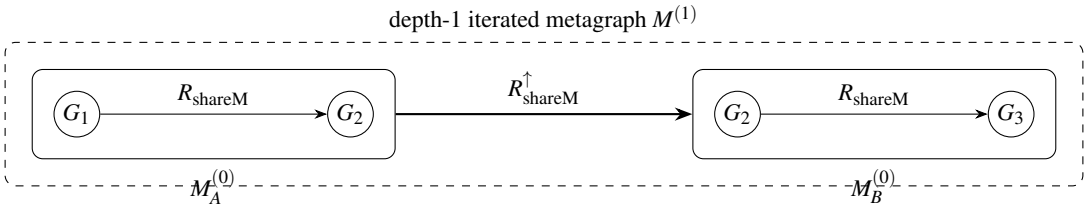


Figure 2. A depth-1 iterated metagraph: vertices are metagraphs $M_A^{(0)}, M_B^{(0)} \in \mathfrak{G}^{(1)}$, and the outer edge is labeled by the lifted relation $R^{\uparrow}_{\text{shareM}} \in \mathscr{R}^{(1)}$.

edges allowed) and

$$\ell : E \to \mathfrak{G}^{\partial 2}$$

is a labeling map. For $e \in E$, writing $\ell(e) = (H_e, p_e^-, p_e^+)$, the intended meaning is: the edge e from $s(e)$ to $t(e)$ carries the internal graph H_e, whose ports p_e^-, p_e^+ will be attached to the endpoints $s(e), t(e)$, respectively.

Definition 2.3 (Expansion / edge-substitution)**.** Let $\mathbb{E} = (V, E, s, t, \ell)$ be an Edge-MetaGraph and write $\ell(e) = (H_e, p_e^-, p_e^+)$. Define a graph Expand($\mathbb{E}$) (the *expansion* of $\mathbb{E}$) by gluing each labeled graph H_e along its ports to the endpoints of e as follows.

Take the disjoint union of vertex sets

$$W \coloneqq V \sqcup \bigsqcup_{e \in E} V(H_e),$$

and let $\sim$ be the smallest equivalence relation on W such that for every $e \in E$,

$$p_e^- \sim s(e), \qquad p_e^+ \sim t(e).$$

Set $V(\text{Expand}(\mathbb{E})) \coloneqq W/\!\sim$. For edges, take the disjoint union

$$F \coloneqq \bigsqcup_{e \in E} E(H_e),$$

and for $f = \{x, y\} \in E(H_e)$ define its endpoints in $V(\text{Expand}(\mathbb{E}))$ to be

$$\big([x]_\sim, [y]_\sim\big).$$

Then Expand($\mathbb{E}$) is the (multi)graph with vertex set $W/\!\sim$ and edge multiset induced from F.

(If $\mathfrak{G}$ is a universe of directed graphs, replace $\{x, y\}$ by ordered pairs (x, y) throughout.)

Example 2.4 (Real-world example: software deployment pipeline with edge-level internal dependencies)**.** Consider a company operating a microservice platform. At a coarse operational level, we describe the *deployment pipeline* as a directed multigraph

$$(V, E, s, t),$$

whose vertices are macro-stages:

$$V = \{\mathsf{Code},\ \mathsf{Build},\ \mathsf{Test},\ \mathsf{Deploy},\ \mathsf{Monitor}\}.$$

A directed edge $e : \mathsf{Build} \to \mathsf{Test}$ represents the handoff "send build artifacts to the test stage". However, this handoff is not atomic: it involves multiple tools and checks (artifact registry, signature verification, test orchestration, reporting).

To model such hidden internal structure, define an Edge–MetaGraph

$$\mathbb{E} = (V, E, s, t, \ell), \qquad \ell : E \to \mathfrak{G}^{\partial 2},$$

where each edge $e \in E$ is labeled by a two-ported internal graph

$$\ell(e) = (H_e, p_e^-, p_e^+).$$

For example, for the edge $e : \mathsf{Build} \to \mathsf{Test}$, take H_e to be the graph whose vertices are

$$V(H_e) = \{\mathsf{ArtifactRepo},\ \mathsf{SignatureCheck},\ \mathsf{TestRunner},\ \mathsf{ReportStore}\},$$

and whose edges represent required interactions (e.g. artifact fetch, signature verification, executing tests, storing reports). Choose ports

$$p_e^- = \mathsf{ArtifactRepo} \in V(H_e),$$
$$p_e^+ = \mathsf{ReportStore} \in V(H_e),$$

so that p_e^- is the entry interface to the internal process (receiving artifacts) and p_e^+ is the exit interface (publishing test reports).

Similarly, for the edge $\mathsf{Deploy} \to \mathsf{Monitor}$, the internal graph $H_{e'}$ may encode

$$\mathsf{DeployController} \to \mathsf{MetricsAgent} \to \mathsf{AlertManager},$$

with ports identifying the start/end of the deployment-to-observability integration.

Figure 3 visualizes the Edge–MetaGraph viewpoint: the outer pipeline is shown together with two illustrative internal graphs attached to edges. By Definition 2.3, the expanded graph Expand($\mathbb{E}$) is obtained by gluing each port p_e^- to $s(e)$ and each port p_e^+ to $t(e)$.

2.2 Iterated Edge-MetaGraphs

An *Iterated Edge-MetaGraph* extends an Edge-MetaGraph recursively by allowing each edge to be labeled by a two-ported Edge-MetaGraph from the preceding level. Consequently, edges may contain nested internal graph structures of arbitrary finite depth, thereby producing a multilevel hierarchy of edge substitution and expansion. We first introduce a general two-port construction for finite vertex-bearing objects.

Definition 2.5 (Two-ported objects)**.** Let $\mathscr{C}$ be a class of finite objects such that each $X \in \mathscr{C}$ is equipped with a finite

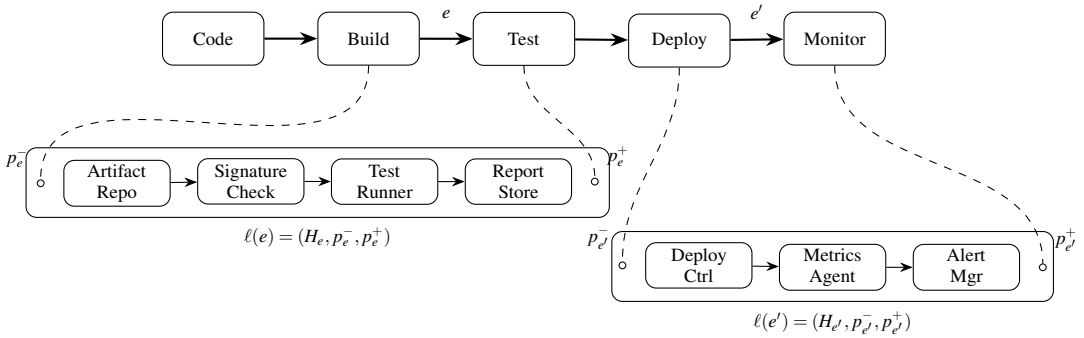


Figure 3. An Edge–MetaGraph for a CI/CD pipeline. The outer directed multigraph encodes macro-stages. Edges e and e' are labeled by two-ported internal graphs; dashed curves indicate the port-gluing used in Expand($\mathbb{E}$).

vertex set $V(X)$. A *two-ported object over* $\mathscr{C}$ is a triple

$$X^{\partial} = (X, p^-, p^+),$$

where

$$X \in \mathscr{C}, \qquad p^-, p^+ \in V(X).$$

The distinguished vertices p^- and p^+ are called, respectively, the *tail-port* and the *head-port*. We write

$$\mathrm{Tail}(X^{\partial}) := p^-, \qquad \mathrm{Head}(X^{\partial}) := p^+,$$

and define

$$\mathscr{C}^{\partial 2} := \left\{ (X, p^-, p^+) \;\middle|\; X \in \mathscr{C},\ p^-, p^+ \in V(X) \right\}.$$

The two ports are not required to be distinct unless explicitly stated otherwise.

Definition 2.6 (Iterated Edge-MetaGraph universes)**.** Let $\mathfrak{G}$ be a nonempty universe of finite graphs. Define recursively a hierarchy

$$\left(\mathfrak{E}^{(k)}\right)_{k \in \mathbb{N}_0}$$

as follows:

$$\mathfrak{E}^{(0)} := \mathfrak{G},$$

and, for every $k \in \mathbb{N}_0$,

$$\mathfrak{E}^{(k+1)} := \left\{ \begin{array}{l} (V, E, s, \tau, \ell) \text{ such that} \\ (V, E, s, \tau) \text{ is a finite} \\ \quad \text{directed multigraph,} \\ \ell : E \longrightarrow \left(\mathfrak{E}^{(k)}\right)^{\partial 2} \end{array} \right\}.$$

Here

$$s, \tau : E \longrightarrow V$$

denote the source and target maps, respectively.

Thus, an element of $\mathfrak{E}^{(k+1)}$ is an Edge-MetaGraph whose every edge is labeled by a two-ported object belonging to the preceding level $\mathfrak{E}^{(k)}$.

Definition 2.7 (Two-ported iterated Edge-MetaGraph)**.** Let $k \in \mathbb{N}_0$. A *two-ported iterated Edge-MetaGraph of depth* k is an element

$$X^{\partial} = (X, p^-, p^+) \in \left(\mathfrak{E}^{(k)}\right)^{\partial 2}.$$

Equivalently,

$$X \in \mathfrak{E}^{(k)}, \qquad p^-, p^+ \in V(X).$$

The vertices p^- and p^+ serve as the external tail and head interfaces of the iterated structure X.

Definition 2.8 (Iterated Edge-MetaGraph of depth k)**.** Let $k \in \mathbb{N}_0$. An *Iterated Edge-MetaGraph of depth* k is an element

$$\mathbb{E}^{(k)} \in \mathfrak{E}^{(k)}.$$

For $k = 0$, an Iterated Edge-MetaGraph is simply a base graph

$$\mathbb{E}^{(0)} \in \mathfrak{G}.$$

For $k \geq 1$, it has the form

$$\mathbb{E}^{(k)} = \left(V^{(k)}, E^{(k)}, s^{(k)}, \tau^{(k)}, \ell^{(k)}\right),$$

where

$$\left(V^{(k)}, E^{(k)}, s^{(k)}, \tau^{(k)}\right)$$

is a finite directed multigraph and

$$\ell^{(k)} : E^{(k)} \longrightarrow \left(\mathfrak{E}^{(k-1)}\right)^{\partial 2}.$$

Hence, for every edge $e \in E^{(k)}$, there exist

$$X_e \in \mathfrak{E}^{(k-1)} \qquad \text{and} \qquad p_e^-, p_e^+ \in V(X_e)$$

such that

$$\ell^{(k)}(e) = (X_e, p_e^-, p_e^+).$$

In particular, a depth-1 Iterated Edge-MetaGraph is precisely an ordinary Edge-MetaGraph whose edges are labeled by two-ported base graphs from $\mathfrak{G}$.

Remark 2.9 (Recursive expansion)**.** The edge-substitution expansion may be extended recursively to Iterated Edge-MetaGraphs.

For a base graph $G \in \mathfrak{E}^{(0)} = \mathfrak{G}$, define

$$\mathrm{Expand}^{(0)}(G) := G,$$

and let

$$\iota_G^{(0)} : V(G) \longrightarrow V\left(\mathrm{Expand}^{(0)}(G)\right)$$

be the identity map.

Suppose recursively that, for every $X \in \mathfrak{E}^{(k)}$, the graph

$\mathrm{Expand}^{(k)}(X)$ and a canonical vertex map

$$\iota_X^{(k)} : V(X) \longrightarrow V\big(\mathrm{Expand}^{(k)}(X)\big)$$

have been defined.

Let

$$\mathbb{E} = (V, E, s, \tau, \ell) \in \mathfrak{E}^{(k+1)},$$

and write

$$\ell(e) = (X_e, p_e^-, p_e^+) \qquad (e \in E).$$

For each $e \in E$, replace the label X_e by its fully expanded graph $\mathrm{Expand}^{(k)}(X_e)$ and use the images

$$\iota_{X_e}^{(k)}(p_e^-), \qquad \iota_{X_e}^{(k)}(p_e^+)$$

as its two attachment ports. Then define

$$\begin{aligned}\mathrm{Expand}^{(k+1)}(\mathbb{E}) := \mathrm{Expand}\big(V, E, s, \tau,\\ e \longmapsto \big(\mathrm{Expand}^{(k)}(X_e),\\ \iota_{X_e}^{(k)}(p_e^-), \iota_{X_e}^{(k)}(p_e^+)\big)\big).\end{aligned}$$

The associated canonical map

$$\iota_{\mathbb{E}}^{(k+1)} : V \longrightarrow V\big(\mathrm{Expand}^{(k+1)}(\mathbb{E})\big)$$

is the quotient-induced map sending each outer vertex to its equivalence class in the expanded graph.

Thus, $\mathrm{Expand}^{(k)}$ recursively removes all nested edge labels down to depth 0 and produces an ordinary graph. The use of the canonical maps $\iota_X^{(k)}$ ensures that the attachment ports remain well defined at every stage of the recursion.

Example 2.10 (Multilevel logistics and service dependencies)**.** Consider an international e-commerce system in which products are transported from a factory to a customer through several logistical stages.

At the top level, consider the directed multigraph

$$\big(V^{(2)}, E^{(2)}, s^{(2)}, \tau^{(2)}\big),$$

with

$$\begin{aligned}V^{(2)} = \{&\mathsf{Factory},\ \mathsf{ExportHub},\\ &\mathsf{ImportHub},\ \mathsf{LastMile},\ \mathsf{Customer}\}.\end{aligned}$$

An edge of $E^{(2)}$ represents a high-level logistical handoff, such as

$$\mathsf{ExportHub} \longrightarrow \mathsf{ImportHub}$$

or

$$\mathsf{LastMile} \longrightarrow \mathsf{Customer}.$$

Such a handoff is generally not an atomic operation. Rather, it is realized through a collection of carriers, checkpoints, customs procedures, and information systems. Accordingly, each top-level edge $e \in E^{(2)}$ is assigned a label

$$\ell^{(2)}(e) = (X_e, p_e^-, p_e^+) \in \big(\mathfrak{E}^{(1)}\big)^{\partial 2},$$

where

$$X_e \in \mathfrak{E}^{(1)}$$

is a depth-1 Edge-MetaGraph describing the internal workflow that realizes the handoff, while p_e^- and p_e^+ specify its entry and exit interfaces.

For example, consider the top-level edge

$$e : \mathsf{ExportHub} \longrightarrow \mathsf{ImportHub}.$$

Its label may contain a depth-1 Edge-MetaGraph X_e with

$$\begin{aligned}V(X_e) = \{&\mathsf{PickUp},\ \mathsf{OriginCustoms},\ \mathsf{Airline},\\ &\mathsf{DestinationCustoms},\ \mathsf{Unload}\}.\end{aligned}$$

The edges of X_e represent operational tasks such as document verification, container inspection, cargo transfer, flight booking, customs clearance, and unloading.

Since $X_e \in \mathfrak{E}^{(1)}$, every task edge $f \in E(X_e)$ is itself labeled by a two-ported base graph:

$$\ell_{X_e}(f) = (H_f, q_f^-, q_f^+) \in \big(\mathfrak{E}^{(0)}\big)^{\partial 2} = \mathfrak{G}^{\partial 2}.$$

For instance, H_f may represent a service-dependency graph of the form

$$\begin{aligned}\mathsf{OrderDB} \longrightarrow \mathsf{RiskScoring} \longrightarrow \mathsf{CustomsAPI}\\ \longrightarrow \mathsf{LabelPrinter},\end{aligned}$$

where q_f^- and q_f^+ identify the entry and exit services associated with the corresponding logistical task.

Consequently, the entire logistics system constitutes an Iterated Edge-MetaGraph of depth 2: the top-level shipment edges are labeled by depth-1 Edge-MetaGraphs, and the edges within those structures are in turn labeled by two-ported base graphs describing service-level dependencies.

Applying the recursive expansion operator $\mathrm{Expand}^{(2)}$ replaces all nested edge labels by their internal structures and yields a single ordinary graph representing the complete factory-to-customer dependency network.

3. CONCLUSION

This paper studied graph-based higher-order structures related to metagraphs and edge-labeled hierarchical networks. In particular, we introduced the concept of an *Edge-MetaGraph*, in which each edge is equipped with a two-ported internal graph, and formalized an edge-substitution expansion obtained by gluing the designated ports to the corresponding edge endpoints. We further developed *Iterated Edge-MetaGraphs*, where edge labels may themselves contain lower-level Edge-MetaGraphs, thereby providing a recursive framework for representing multilevel and nested network structures. The illustrative examples involving software deployment pipelines and logistics systems demonstrated how the proposed framework can encode internal dependencies that are hidden at the level of an ordinary graph.

As directions for future research, further extensions incorporating uncertainty-oriented graph models, such as *Fuzzy Graphs*, *Intuitionistic Fuzzy Graphs*[19], *Neutrosophic Graphs*[4], and *Plithogenic Graphs*[20], may be investigated. It would also be useful to study structural properties of Edge-MetaGraphs and Iterated Edge-MetaGraphs, including connectivity, paths, cycles, reachability, and matrix-based representations. In addition, the design and analysis of efficient

graph algorithms for construction, recursive expansion, traversal, and structural analysis of these generalized networks constitute promising topics for future work.

ACKNOWLEDGMENTS

We extend our sincere gratitude to everyone who provided insights, inspiration, and assistance throughout this research. We particularly thank our readers for their interest and acknowledge the authors of the cited works for laying the foundation that made our study possible. We also appreciate the support from individuals and institutions that provided the resources and infrastructure needed to produce and share this paper. Finally, we are grateful to all those who supported us in various ways during this project.

REFERENCES

[1] J. L. Gross, J. Yellen, and M. Anderson, *Graph theory and its applications*, 3rd ed. Chapman and Hall/CRC, 2018.

[2] A. Rosenfeld, "Fuzzy graphs," in *Fuzzy sets and their applications to cognitive and decision processes*, L. A. Zadeh, K.-S. Fu, K. Tanaka, and M. Shimura, Eds. Academic Press, 1975, pp. 77–95.

[3] J. N. Mordeson and P. S. Nair, *Fuzzy graphs and fuzzy hypergraphs*, ser. Studies in Fuzziness and Soft Computing. Physica-Verlag, 2000, vol. 46.

[4] M. Alqahtani, M. Kaviyarasu, A. Al-Masarwah, and M. Rajeshwari, "Application of complex neutrosophic graphs in hospital infrastructure design," *Mathematics*, vol. 12, no. 5, p. 719, 2024.

[5] S. Yaseen, A. Salih, and M. Alomari, "Adding neutrosophic topological indices to graphs for social network analysis," *Journal of Mathematics*, vol. 2026, no. 1, p. 3772943, 2026.

[6] M. Rajeshwari, M. Kaviyarasu, and M. Alqahtani, "Topological analysis of iot systems using uniform neutrosophic graphs," *Ain Shams Engineering Journal*, vol. 17, no. 4, p. 104079, 2026.

[7] R. B. Lima Filho, R. Sabino-Silva, and M. G. Carneiro, "High-level classification based on meta-graph of visibility graphs for autism detection," in *2025 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2025, pp. 1–8.

[8] R. B. R. Azevedo, R. Lohaus, and T. Paixão, "Networking networks," *Evolution & Development*, vol. 10, no. 5, pp. 514–515, 2008.

[9] E. Velazquez-Garcia, I. Lopez-Arevalo, and V. J. Sosa-Sosa, "Representing document semantics by means of graphs," in *2011 8th International Conference on Electrical Engineering, Computing Science and Automatic Control*. IEEE, 2011, pp. 1–6.

[10] S. Ciliberti, O. C. Martin, and A. Wagner, "Innovation and robustness in complex regulatory gene networks," *Proceedings of the National Academy of Sciences*, vol. 104, no. 34, pp. 13 591–13 596, 2007.

[11] Y. Schaerli, A. Munteanu, M. Gili, J. Cotterell, J. Sharpe, and M. Isalan, "A unified design space of synthetic stripe-forming networks," *Nature communications*, vol. 5, no. 1, p. 4905, 2014.

[12] C. Donnat and S. Holmes, "Tracking network dynamics: A survey using graph distances," *The Annals of Applied Statistics*, vol. 12, no. 2, pp. 971–1012, 2018.

[13] A. Bretto, *Hypergraph Theory: An Introduction*, ser. Mathematical Engineering. Springer, 2013.

[14] G. Karypis, "Multilevel hypergraph partitioning," in *Multilevel Optimization in VLSICAD*, ser. Combinatorial Optimization, J. Cong and J. R. Shinnerl, Eds. Springer, 2003, vol. 14, pp. 125–154.

[15] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 3558–3565.

[16] K. Lalithambigai, P. Gnanachandra, and S. Jafari, "Dual perspective on adjacency relation and fuzzy coloring in fuzzy super hypergraphs," in *SuperHyperTopologies and SuperHyperStructures with their Applications: Collected Papers*, F. Smarandache, G. Nordo, and S. Jafari, Eds. Infinite Study, 2025, pp. 82–96.

[17] F. Smarandache, "Extension of hypergraph to n-superhypergraph and to plithogenic n-superhypergraph, and extension of hyperalgebra to n-ary (classical-/neutro-/anti-) hyperalgebra," *Neutrosophic Sets and Systems*, vol. 33, pp. 290–296, 2020.

[18] T. Fujita and F. Smarandache, "Representing higher-order networks: A survey of graph-based frameworks," 2026, arXiv:2605.12509.

[19] M. G. Karunambigai, R. Parvathi, and R. Buvaneswari, "Arc in intuitionistic fuzzy graphs," *Notes on Intuitionistic Fuzzy Sets*, vol. 17, no. 4, pp. 37–47, 2011.

[20] F. Sultana, M. Gulistan, M. Ali, N. Yaqoob, M. Khan, T. Rashid, and T. Ahmed, "A study of plithogenic graphs: applications in spreading coronavirus disease (covid-19) globally," *Journal of ambient intelligence and humanized computing*, vol. 14, no. 10, pp. 13 139–13 159, 2023.