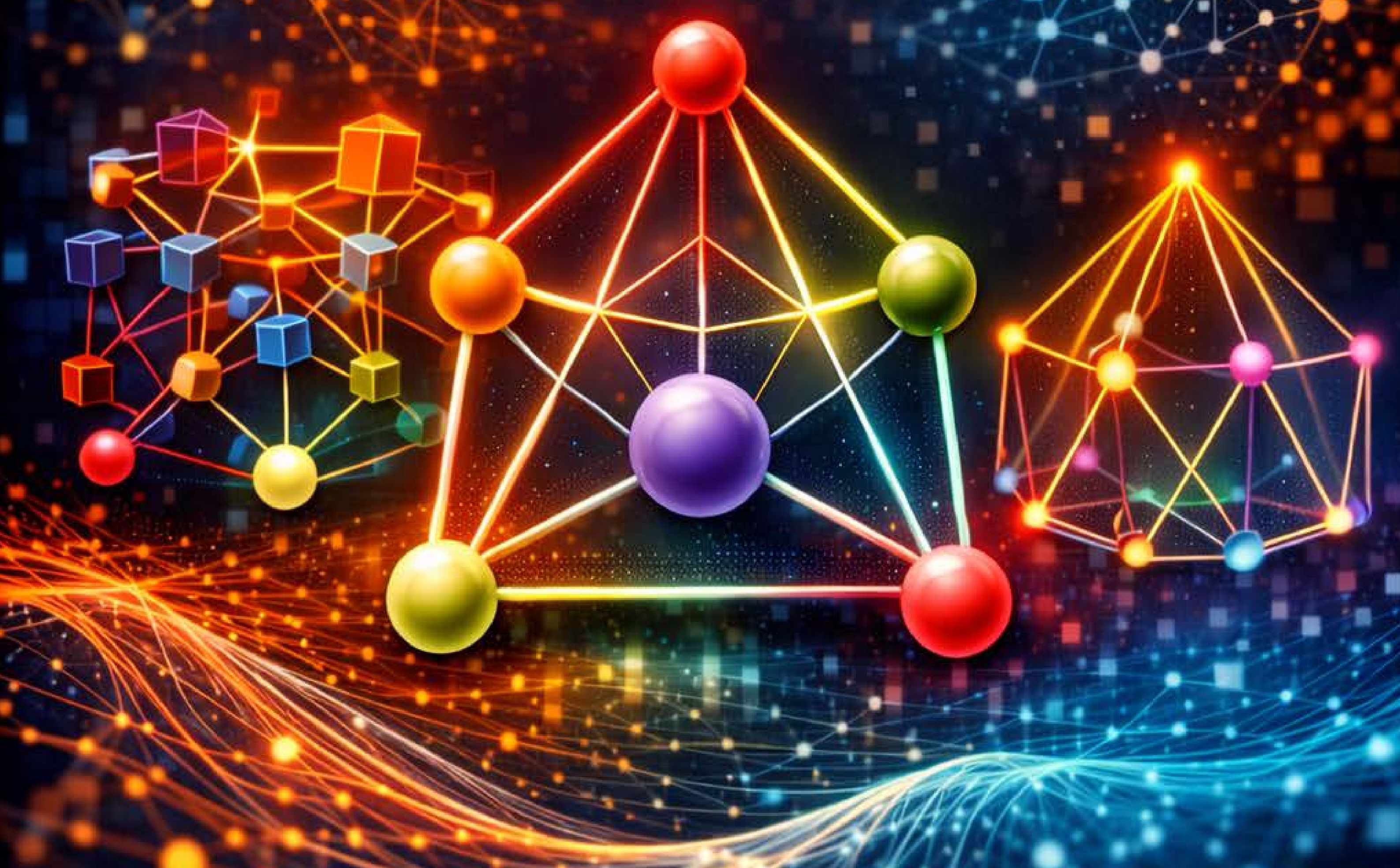
Representing HIGHER-ORDER NETWORKS

A Survey of Graph-Based Frameworks

Third Edition



Takaaki Fujita
Florentin Smarandache

Takaaki Fujita, Florentin Smarandache

Representing Higher-Order Networks: A Survey of Graph-Based Frameworks

(Third Edition)

Neutrosophic Science International Association (NSIA)
Publishing House

Gallup - Guayaquil
United States of America – Ecuador
2026

Editor:

Neutrosophic Science International Association (NSIA)
Publishing House
https://fs.unm.edu/NSIA/

Division of Mathematics and Sciences
University of New Mexico
705 Gurley Ave., Gallup Campus
NM 87301, United States of America

University of Guayaquil
Av. Kennedy and Av. Delta
"Dr. Salvador Allende" University Campus
Guayaquil 090514, Ecuador

PEER REVIEWERS

Mohamed Elhoseny
American University in the Emirates, Dubai, UAE
Email: mohamed.elhoseny@aue.ae

Young Bae Jun
Gyeongsang National University, South Korea
Email: skywine@gmail.com

Yo-Ping Huang
Department of Computer Science and Information, Engineering
National Taipei University, New Taipei City, Taiwan
Email: yphuang@ntut.edu.tw

ISBN 978-197250202-0

Table of Contents

1 Introduction

1.1 Higher-Order Graphs

It is well known that many real-world phenomena can be modeled using graphs and networks [1, 2]. However, many such systems exhibit structures that go beyond pairwise interactions: they may involve multiway relations, hierarchical organization, nested or recursive dependencies, temporal evolution, or multilayer coupling. Classical graph models are often insufficient to represent these features in a mathematically faithful way.

To address this limitation, a wide range of higher-order formalisms has been developed, including hypergraphs [3], superhypergraphs [4], metagraph-based models [5], simplicial and cell-complex-based frameworks, multilayer and temporal networks, and more recent category-theoretic or semantic approaches. For instance, superhypergraphs extend higher-order network models by allowing the *vertex domain itself* to be hierarchical, thereby enabling set-valued and iterated structures to be encoded directly in the object domain [4]. As a result, higher-order graph theory has grown into a broad and heterogeneous landscape, with many concepts arising from different mathematical viewpoints and modeling goals (cf. [6, 7, 8, 9, 10, 11]).

The concept of Higher-Order Networks (or Higher-Order Structures) has been applied in fields such as the following. Of course, the range of applications is not limited to these alone:

- **Transportation and logistics networks (cf. [12, 13, 14, 15, 16]):** Higher-order graphs can represent multi-stop delivery plans, hub coordination, shared routes, and group-wise flow constraints more naturally than ordinary pairwise graphs.
- **Social network analysis (cf. [17, 18, 19, 20, 21, 22]):** They model group conversations, team interactions, overlapping communities, and hierarchical memberships, going beyond simple person-to-person links.
- **Knowledge representation and semantic networks (cf. [23, 24]):** They are useful for encoding multi-entity relations, typed facts, contextual associations, and hierarchical semantic structures in knowledge systems.
- **Molecular and chemical structure analysis (cf. [25, 26, 27]):** They can describe multi-atom interactions, reaction mechanisms, molecular complexes, and higher-order structural dependencies in chemical systems.
- **Neuroscience and brain networks (cf.[28, 29]):** They support the modeling of collective neural interactions, multi-region synchronization, layered brain connectivity, and time-dependent functional organization.
- **Machine learning and graph neural networks (cf. [30, 31, 32, 10, 33]):** Higher-order graphs are applied to learning tasks where relations involve groups, hierarchies, or nested structures, such as HyperGraph Neural Networks and related models.
- **Recommendation systems (cf.[34, 35, 36]):** They can simultaneously capture users, items, contexts, time, and attribute interactions, providing richer relational representations than ordinary bipartite graphs.
- **Supply chains and organizational systems (cf.[37]):** They model multi-party dependencies among suppliers, resources, departments, and processes, including hierarchical and cross-level coordination structures.
- **Communication and information networks (cf.[38]):** They are suitable for representing multicast communication, layered protocols, group transmission, and dynamically changing higher-order connectivity patterns.
- **Decision-making and operations research (cf.[39, 40, 41, 42]):** They can express interacting criteria, grouped alternatives, hierarchical evaluation structures, and uncertainty-aware relational dependencies in complex decision problems.

1.2 Our Contributions

A wide variety of mathematical frameworks for representing higher-order networks has already been developed. However, these frameworks are often dispersed across different mathematical traditions, terminologies, and application areas, which makes systematic comparison difficult. For this reason, we consider it valuable to compile a survey-style book that brings these concepts together within a single coherent reference.

Accordingly, this book provides a broad and structured overview of mathematical notions that can be used to model higher-order networks. Its purpose is to offer a unified point of entry to these formalisms, to clarify their foundational ideas, and to highlight both their common features and their essential differences. In this way, the book is intended to support further theoretical development as well as applications in areas such as AI and related disciplines.

It is important to note, however, that the concepts collected here are not “higher-order” in one single uniform sense. Some frameworks generalize graphs by increasing the arity of interactions, as in hypergraph-type models. Others introduce hierarchy, nesting, or recursion, as in superhypergraph-type constructions. Still others encode higher-orderness through layers, temporal indexing, or multi-aspect organization, as in multilayer and temporal networks. Finally, some approaches arise from different mathematical semantics altogether, including operadic, monoidal, relational, tensor-based, closure-based, and coalgebraic viewpoints.

To make this diversity easier to understand and compare, the concepts in this book are organized into four broad families, in accordance with the practical classification adopted in the summary tables:

1. combinatorial, set-theoretic, and order-theoretic structures,
2. geometric, topological, and complex-based structures,
3. factorization-, constraint-, layered-, temporal-, and tensor-based structures, and
4. semantic, compositional, knowledge-based, and logical structures.

As a reference, a practical four-family organization of higher-order network concepts used in this book is provided in Table 1.1. This classification is not based on the mathematical nature of the objects themselves, but rather on a practical classification according to their principal organizing viewpoint.

Table 1.1: A practical four-family organization of higher-order network concepts used in this book.

Family	**Main organizing viewpoint**	**Representative concepts in this book**
I. Combinatorial, set-theoretic, and order-theoretic family	This family emphasizes higher-order structure arising from combinatorial incidence, set systems, containment, recursion, iteration, hierarchical membership, algebraic hyperoperations, and order-based constructions.	Hyperstructure and Superhyperstructure; HyperGraph and SuperHyperGraph; MultiGraph and Iterated MultiGraph; h-model; Chain-Free Subsets; Power Set Graph; Johnson Graph; Kneser Graph; Meta-Graph and Iterated Meta-Graph; Meta-HyperGraph and Meta-SuperHyperGraph; Nested HyperGraph and Nested SuperHyperGraph; Multi-Hypergraph and Multi-SuperHypergraph; Line Graph and Iterated Line Graph; Iterated Total Graph; Hierarchical SuperHyperGraph; Recursive HyperGraph and Recursive SuperHyperGraph; Tree-Vertex Graph; MultiMeta-Graph; Transfinite SuperHyperGraph; Graded superhypergraph; HyperMatroid; SuperHyperMatroid; Kneser SuperHypergraphs; Iterated Multi-Edge Graph; Iterated Multi-Recursive Graph; Multi-Axis SuperHyperGraph.

Continued on the next page

Table 1.1 (continued)

Family	Main organizing viewpoint	Representative concepts in this book
II. Geometric, topological, and complex-based family	This family organizes higher-order networks through simplices, cells, cubes, polyhedra, paths, sheaf-like local-to-global structures, topological realizations, refinement procedures, and related geometric incidence frameworks.	Abstract simplicial complex; Simplicial set; Cell complex; CW complex; Polyhedral complex; Dowker Complex; Cubical Complex; Path Complex; Cellular Sheaf; Meta Simplicial Complex; Simplicial SuperHypercomplex; Depth-r iterated subdivisions of polyhedral complexes; Topological SuperHyperGraph.
III. Factorization, constraint, layered, temporal, and tensor-based family	This family focuses on higher-order structure induced by factorization, coding constraints, layer/time indexing, product-state organization, and tensorial interaction encoding.	Factor graph; Tanner graph; Tanner Hypergraph; Tanner SuperHyperGraph; Multilayer network; Temporal network; MultiDimensional Graph; Tensor network graph; MultiTensor and Iterated MultiTensor Network; Tensor Hypernetwork and Tensor Superhypernetwork; Tensor Train; Tree Tensor Network (TTN); Projected Entangled Pair State (PEPS); Projected Entangled Simplex State (PESS); Adjacency-Tensor Network (ATN).
IV. Semantic, compositional, knowledge, and logical family	This family treats higher-order networks through meaning, typing, composition, interfaces, transformation, knowledge representation, logical implication, uncertainty, and other semantic enrichments.	Open Hypergraph and Open SuperHyperGraph; Heterogeneous Graph, HyperGraph, and SuperHyperGraph; Knowledge Graph, HyperGraph, and SuperHyperGraph; Petri Net; Port Graph; Port HyperGraph and Port SuperHyperGraph; Combinatorial Map; Cognitive HyperGraphs and Cognitive SuperHyperGraphs; Multimodal Graph, HyperGraph, and SuperHyperGraph; Operadic Interaction Graph (OIG); Symmetric Monoidal Wiring Graph (SMWG); Relational-Arity Graph (RAG); Closure-Implication Graph (CIG); Coalgebraic Nested-Neighborhood Graph (CNNG); Curried Graph; Sheaf HyperGraph / Sheaf SuperHyperGraph; Fibered HyperGraph / Fibered SuperHyperGraph; Galois HyperGraph / Galois SuperHyperGraph; Rewrite HyperGraph / Rewrite SuperHyperGraph; Uncertain SuperHyperGraph; Functorial SuperHyperGraph; Motif Hypergraphs and Motif SuperHypergraphs; Molecular SuperHyperGraphs.

This book is Edition 3.0. It mainly includes the addition of several concepts, as well as corrections and improvements of typographical errors and explanations. Compared with Edition 1.0, it contains substantial additions.

Representing Higher-Order Networks: A Survey of Graph-Based Frameworks (Third Edition)

Takaaki Fujita [1] * **and Florentin Smarandache**[2]
[1] Independent Researcher, Tokyo, Japan.
Email: Takaaki.fujita060@gmail.com
[2] University of New Mexico, Gallup Campus, NM 87301, USA.
Email: fsmarandache@gmail.com

Abstract

Many real-world phenomena are naturally modeled by graphs and networks. However, classical graph models are often limited to pairwise interactions and may not adequately capture the richer structures that arise in practice. Higher-order graph formalisms extend this framework by incorporating multiway, hierarchical, temporal, multilayer, recursive, and tensor-based interactions, thereby providing more expressive representations of complex systems.

This book presents a comprehensive overview of mathematical notions that can be used to model higher-order networks. It surveys foundational concepts, extensional frameworks, and newly introduced formalisms, with an emphasis on their structural principles, relationships, and modeling roles. The aim is to provide a unified perspective that helps readers compare diverse higher-order network models and identify appropriate tools for theoretical study and practical applications.

This book is Edition 3.0. It mainly includes the addition of several concepts, as well as corrections and improvements of typographical errors and explanations. Compared with Edition 1.0, it contains substantial additions.

Keywords: Hypergraph, Superhypergraph, Higher-Order Graphs

MSC2010 (Mathematics Subject Classification 2010): 05C65 - Hypergraphs, 05C82 - Graph theory with applications

2 Combinatorial, set-theoretic, and order-theoretic family

In this chapter, we describe the main types of higher-order graphs. For reference, the combinatorial, set-theoretic, and order-theoretic higher-order structures treated in this book are listed in Table 2.1.

Table 2.1: Combinatorial, set-theoretic, and order-theoretic higher-order structures treated in this book.

Concept	**Concise description**
HyperGraph and SuperHyperGraph	Set-based higher-order structures modeling multiway relations and hierarchical supervertices through iterated powerset constructions.
MultiGraph and Iterated MultiGraph	Graph structures with multiplicities, extended to iterated multiset-based vertices for recursive higher-order organization.
h-model	A logical hypergraph-based semantic framework assigning hypergraphs to propositional atoms over a common base set.
Chain-Free Subsets	Order-theoretic structures built from subsets avoiding long chains, capturing combinatorial incomparability patterns.
Power Set Graph	Graphs whose vertices are subsets and whose adjacency is determined by inclusion relations.
Johnson Graph	Graphs on fixed-cardinality subsets, encoding adjacency by single-element replacement.
Kneser Graph	Graphs on fixed-cardinality subsets, encoding adjacency by disjointness.
Meta-Graph and Iterated Meta-Graph	Graphs whose vertices are graphs, iterated recursively to represent graph-of-graphs organization.
Meta-HyperGraph and Meta-SuperHyperGraph	Hypergraph-style higher-order structures whose vertices are hypergraphs or superhypergraphs themselves.
Nested HyperGraph and Nested SuperHyperGraph	Structures allowing edges to contain lower-level edges, yielding well-founded nested incidence.
Multi-Hypergraph and Multi-SuperHypergraph	Hypergraph and superhypergraph models with repeated hyperedges or superhyperedges via multiplicity.
Line Graph and Iterated Line Graph	Edge-incidence transformations turning edges into vertices and iterating this process recursively.
Iterated Total Graph	Repeated total-graph constructions encoding both adjacency and incidence across multiple levels.
Hierarchical SuperHyperGraph	Superhypergraphs with mixed-level vertices and coherence across powerset layers.
Recursive HyperGraph and Recursive SuperHyperGraph	Hypergraph-type structures whose edges may recursively contain lower-level edges or superedges.
Tree-Vertex Graph	Rooted hierarchical structures whose vertices are organized through nested labels on a tree.
Tensor network graph	Graph-based representations of tensor contractions encoding multiway algebraic interactions combinatorially.
MultiTensor and Iterated MultiTensor Network	Tensor-network models assigning finite multisets or iterated finite multisets of local tensors to vertices, yielding multiple weighted contractions through realization choices and recursive flattening.

Continued on the next page

Table 2.1 (continued)

Concept	Concise description
Tensor Hypernetwork and Tensor Superhypernetwork	Hypergraph- and superhypergraph-based tensor networks in which multiway or hierarchical incidence structures guide simultaneous tensor contractions.
Tensor Train	A tensor decomposition expressing a high-order tensor as a chain of third-order core tensors linked by auxiliary bond indices.
Tree Tensor Network (TTN)	A loop-free hierarchical tensor decomposition on a tree, representing many-body states through local tensors and virtual bonds.
Projected Entangled Pair State (PEPS)	A tensor-network state obtained by projecting entangled virtual edge pairs onto physical lattice sites, encoding many-body correlations.
Projected Entangled Simplex State (PESS)	A tensor-network state assigning entangled virtual tensors to simplices and projecting them onto physical sites, generalizing PEPS from edges to simplices.
MultiMeta-Graph	A graph whose vertices are finite families of graphs rather than single graph objects.
Transfinite SuperHyperGraph	Superhypergraph structures extended across ordinally indexed transfinite levels.
Multi-Axis SuperHyperGraph	Superhypergraphs organized along several independent powerset axes with multi-indexed levels.
Iterated Multi-Edge Graph	Graph structures whose edge objects are iterated multisets of endpoint pairs.
Iterated Multi-Recursive Graph	Recursive graph-like structures combining iterated multiset vertices and iterated multiset edges.
HyperMatroid	Circuit-based dependence structures viewed as hypergraph-like higher-order combinatorial systems.
SuperHyperMatroid	Matroid-like dependence structures defined on supervertices and supercircuits.
Kneser SuperHypergraphs	Superhypergraph extensions of Kneser-type disjointness constructions via flattened supports.
Graded superhypergraph	Superhypergraphs whose vertices carry explicit grades or levels across powerset hierarchy.
Hyperstructure and Superhyperstructure	Algebraic higher-order structures based on set-valued operations and their iterated powerset extensions for modeling hierarchical multi-level interactions.

2.1 HyperGraph and SuperHyperGraph

Superhypergraphs extend higher–order network models by allowing the *vertex domain itself* to be hierarchical. Concretely, one starts from a base set and iterates the powerset operation; vertices (often called *supervertices*) may then be set-valued objects living at a prescribed level of this iteration, while (super)edges encode incidence among these higher-level vertices [4]. Related hierarchical constructions have been explored in applications [43].

In addition, several extensions of hypergraphs are known, including fuzzy hypergraphs[44, 45, 46], neutrosophic hypergraphs[47, 48, 49], and plithogenic hypergraphs [50]. Likewise, extensions of superhypergraphs such as fuzzy superhypergraphs [41], neutrosophic superhypergraphs [51, 52], and plithogenic superhypergraphs [53, 54, 55] have been studied. Additionally, as oriented graph concepts, the following are known: Directed HyperGraph[56, 57], Bidirected HyperGraph[58], Directed SuperHyperGraph [59, 60], Oriented Hypergraph[61, 62, 63, 64], Oriented SuperHypergraph, and Bidirected SuperHyperGraph [65, 58]. For a broader overview, we refer the reader to the survey monograph [66].

Definition 2.1.1 (Iterated powerset)**.** [67] For $k \in \mathbb{N}_0$, define iterated powersets recursively by

$$\mathcal{P}^0(X) := X, \qquad \mathcal{P}^{k+1}(X) := \mathcal{P}\big(\mathcal{P}^k(X)\big).$$

For the nonempty variant, set

$$(\mathcal{P}^*)^0(X) := X, \qquad (\mathcal{P}^*)^{k+1}(X) := \mathcal{P}^*((\mathcal{P}^*)^k(X)),$$

where $\mathcal{P}^*(Y) := \mathcal{P}(Y) \setminus \{\varnothing\}$.

Definition 2.1.2 (Hypergraph)**.** [3, 68] A *hypergraph* is a pair $H = (V(H), E(H))$ such that $V(H) \neq \varnothing$ and

$$E(H) \subseteq \mathcal{P}^*(V(H)).$$

Throughout this book, both $V(H)$ and $E(H)$ are assumed finite.

Example 2.1.3 (Team-based bug triage as a hypergraph)**.** Let $V(H)$ be a set of software engineers in a company,

$$V(H) = \{\mathsf{Alice},\ \mathsf{Bob},\ \mathsf{Chen},\ \mathsf{Dina},\ \mathsf{Eli}\}.$$

Each bug report is typically triaged by a *group* (not just a pair), e.g. a reviewer, a domain expert, and a release manager. We model each triage group as a hyperedge. For instance, set

$$E(H) = \big\{\{\mathsf{Alice}, \mathsf{Bob}, \mathsf{Chen}\},\ \{\mathsf{Bob}, \mathsf{Dina}\},\ \{\mathsf{Chen}, \mathsf{Dina}, \mathsf{Eli}\},\ \{\mathsf{Alice}, \mathsf{Eli}\}\big\} \subseteq \mathcal{P}^*(V(H)).$$

Then $H = (V(H), E(H))$ is a finite hypergraph in the sense of Definition 2.1.2, where each hyperedge represents a multi-person triage interaction.

Definition 2.1.4 (n-SuperHyperGraph)**.** [4, 69] Fix a finite base set V_0 and an integer $n \in \mathbb{N}_0$. An *n-SuperHyperGraph over V_0* is a triple

$$\mathrm{SHG}^{(n)} = (V, E, \partial),$$

where

- $V \subseteq \mathcal{P}^n(V_0)$ is a finite set of *n-supervertices*;
- E is a finite set of *(super)edge identifiers*;
- $\partial : E \to \mathcal{P}^*(V)$ is an *incidence map* such that $\partial(e) \subseteq V$ is a nonempty finite set for every $e \in E$.

The set $\partial(e)$ is called the *incidence set* (or *superincidence set*) of e.

Example 2.1.5 (Two-level organization chart as a 1-SuperHyperGraph)**.** Let the base set list individual employees:

$$V_0 = \{a, b, c, d, e, f\}.$$

Consider *teams* as 1-supervertices (subsets of V_0). Define

$$V = \big\{\{a, b\},\ \{c, d, e\},\ \{f\}\big\} \subseteq \mathcal{P}^1(V_0) = \mathcal{P}(V_0).$$

Now define a set of superedge identifiers $E = \{e_1, e_2\}$ and the incidence map

$$\partial : E \to \mathcal{P}^*(V)$$

by

$$\partial(e_1) = \big\{\{a, b\}, \{c, d, e\}\big\}, \qquad \partial(e_2) = \big\{\{c, d, e\}, \{f\}\big\}.$$

Then $\mathrm{SHG}^{(1)} = (V, E, \partial)$ is a 1-SuperHyperGraph over V_0 in the sense of Definition 2.1.4. Here e_1 encodes a collaboration between the two teams $\{a, b\}$ and $\{c, d, e\}$, while e_2 encodes a coordination link between the team $\{c, d, e\}$ and the individual team $\{f\}$. A reference illustration for this example is provided in Fig. 2.1.

Example 2.1.6 (A 2-SuperHyperGraph with nested supervertices)**.** Let the base set be

$$V_0 = \{a, b, c\}.$$

For $n = 2$, we have $\mathcal{P}^2(V_0) = \mathcal{P}(\mathcal{P}(V_0))$, so a 2-supervertex is a *set of subsets* of V_0. Define the 2-supervertex set

$$V = \Big\{v_1 = \{\{a\}, \{a, b\}\},\quad v_2 = \{\{b\}, \{b, c\}\},\quad v_3 = \{\{c\}\}\Big\} \ \subseteq\ \mathcal{P}^2(V_0).$$

Let the superedge identifier set be

$$E = \{e_1, e_2\},$$

and define the incidence map $\partial : E \to \mathcal{P}^*(V)$ by

$$\partial(e_1) = \{v_1, v_2\}, \qquad \partial(e_2) = \{v_2, v_3\}.$$

Then $\mathrm{SHG}^{(2)} = (V, E, \partial)$ is a 2-SuperHyperGraph over V_0 in the sense of Definition 2.1.4. Here e_1 links the two nested supervertices v_1 and v_2, while e_2 links v_2 and v_3.

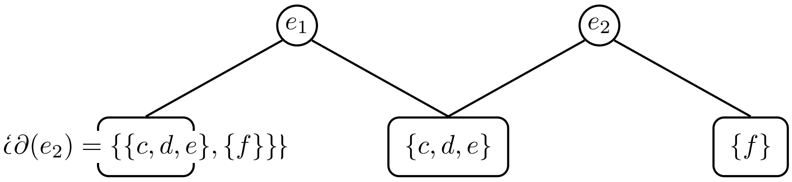


Figure 2.1: A two-level organization chart modeled as a 1-SuperHyperGraph $\mathrm{SHG}^{(1)} = (V, E, \partial)$. Teams are 1-supervertices, and e_1, e_2 are superedges with incidence given by ∂.

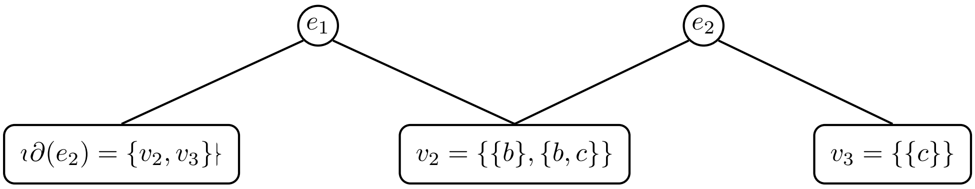


Figure 2.2: A 2-SuperHyperGraph $\mathrm{SHG}^{(2)} = (V, E, \partial)$ over $V_0 = \{a, b, c\}$. Each 2-supervertex is a set of subsets of V_0, and e_1, e_2 are superedges with incidence given by ∂.

A (m, n)-SuperHyperGraph is a mathematical structure in which each vertex corresponds to an (m, n)-superhyperfunction defined on a base set, while the hyperedges group such functions together to represent higher-order relationships and contextual connections. An (h, k)-ary (m, n)-SuperHyperGraph further generalizes this idea by taking vertices as (h, k)-ary (m, n)-superhyperfunctions.

Notation 2.1.7. For a nonempty base set S define

$$\mathcal{P}_0(S) := S, \qquad \mathcal{P}_{m+1}(S) := \mathcal{P}\big(\mathcal{P}_m(S)\big) \quad (m \in \mathbb{N}_0),$$

so $\mathcal{P}_1(S) = \mathcal{P}(S)$, $\mathcal{P}_2(S) = \mathcal{P}(\mathcal{P}(S))$, etc. We also use the Cartesian power $X^h := \underbrace{X \times \cdots \times X}_{h \text{ copies}}$ for $h \in \mathbb{N}$.

Definition 2.1.8 (*(m, n)-superhyperfunction*)**.** [70, 71] Let $m, n \in \mathbb{N}$ and $S \neq \varnothing$. An *(m, n)-superhyperfunction on S* is a map

$$f : \mathcal{P}_m(S) \longrightarrow \mathcal{P}_n(S).$$

Equivalently, $f \in \mathrm{Hom}\big(\mathcal{P}_m(S), \mathcal{P}_n(S)\big)$ as functions of sets.

Definition 2.1.9 (*(m, n)-SuperHyperGraph*)**.** Fix $m, n \in \mathbb{N}$ and a nonempty base set S. Let

$$\mathfrak{F}_{m,n}(S) := \Big\{ f : \mathcal{P}_m(S) \to \mathcal{P}_n(S) \Big\}.$$

An (m, n)-*SuperHyperGraph* is a pair

$$\mathsf{SHG}^{(m,n)} := (V, \mathcal{E}),$$

where $V \subseteq \mathfrak{F}_{m,n}(S)$ is a nonempty set of vertices (each vertex is a concrete (m, n)-superhyperfunction) and

$$\varnothing \neq \mathcal{E} \subseteq \mathcal{P}(V) \setminus \{\varnothing\}$$

is a nonempty family of nonempty *hyperedges*. Each hyperedge $E \in \mathcal{E}$ groups a finite, nonempty set of superhyperfunctions to encode higher-order relations/constraints among them.

Example 2.1.10 (A concrete $(2, 1)$-SuperHyperGraph on $S = \{a, b\}$)**.** Let the base set be

$$S = \{a, b\}.$$

Then

$$\mathcal{P}_1(S) = \mathcal{P}(S) = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}, \qquad \mathcal{P}_2(S) = \mathcal{P}(\mathcal{P}(S)).$$

We construct a small family of $(2, 1)$-superhyperfunctions, i.e. maps

$$f : \mathcal{P}_2(S) \to \mathcal{P}_1(S) = \mathcal{P}(S).$$

Table 2.2: A concise comparison of graphs, hypergraphs, n-superhypergraphs, and (m, n)-superhypergraphs.

Structure	**Vertex domain**	**Edge / incidence object**	**What it captures (one line)**
Graph $G = (V, E)$	V is a (finite) set of *atomic* vertices	$E \subseteq \mathcal{P}_2^*(V) = \{\{u, v\} \subseteq V : u \neq v\}$ (pairwise edges)	Pairwise interactions only (binary relations).
Hypergraph $H = (V, E)$	V is a (finite) set of atomic vertices	$E \subseteq \mathcal{P}^*(V)$ (hyperedges are nonempty subsets of V)	Multiway interactions among arbitrary-size groups.
n-SuperHyperGraph $\mathrm{SHG}^{(n)} = (V, E, \partial)$	$V \subseteq \mathcal{P}^n(V_0)$ (vertices are n-*level* set-valued objects over a base set V_0)	E is a set of edge identifiers and $\partial : E \to \mathcal{P}^*(V)$ (incidence among supervertices)	Higher-order interactions *and* hierarchical/nested vertex semantics via iterated powersets.
(m, n)-SuperHyperGraph $\mathsf{SHG}^{(m,n)} = (V, \mathcal{E})$	$V \subseteq \mathfrak{F}_{m,n}(S) = \{f : \mathcal{P}_m(S) \to \mathcal{P}_n(S)\}$ (vertices are *superhyperfunctions*)	$\varnothing \neq \mathcal{E} \subseteq \mathcal{P}(V) \setminus \{\varnothing\}$ (hyperedges group such functions)	Higher-order relations among *operators* between hierarchical domains/codomains (contextual constraints on maps).

Define three functions $f_1, f_2, f_3 \in \mathfrak{F}_{2,1}(S)$ as follows. For any $X \in \mathcal{P}_2(S) = \mathcal{P}(\mathcal{P}(S))$ (so X is a set of subsets of S), set

$$f_1(X) := \bigcup_{A \in X} A,$$

(the union of all subsets in X),

$$f_2(X) := \bigcap_{A \in X} A,$$

(with the convention $\bigcap \emptyset = S$), and

$$f_3(X) := \begin{cases} \{a\}, & \text{if } \{a\} \in X, \\ \emptyset, & \text{otherwise.} \end{cases}$$

Let the vertex set be the nonempty set of concrete $(2, 1)$-superhyperfunctions

$$V = \{f_1, f_2, f_3\} \subseteq \mathfrak{F}_{2,1}(S).$$

Define a nonempty family of hyperedges by

$$\mathcal{E} = \{E_1, E_2\} \subseteq \mathcal{P}(V) \setminus \{\emptyset\}, \qquad E_1 = \{f_1, f_2\}, \quad E_2 = \{f_2, f_3\}.$$

Then

$$\mathsf{SHG}^{(2,1)} = (V, \mathcal{E})$$

is a concrete $(2, 1)$-SuperHyperGraph in the sense of Definition 2.1.9. Here E_1 groups the "aggregation" superhyperfunctions f_1 (union) and f_2 (intersection), while E_2 groups f_2 with the indicator-type superhyperfunction f_3, representing a different contextual relationship among superhyperfunctions.

For reference, a comparison of graphs, hypergraphs, n-superhypergraphs, and (m, n)-superhypergraphs is provided in Table 2.2.

2.2 MultiGraph and Iterated MultiGraph

A multigraph is a graph allowing parallel edges and loops; formally edges are a multiset with multiplicities between vertices possibly [72, 73, 74]. As extensions, concepts such as fuzzy multigraphs [75, 76, 77], bipartite multigraphs[78, 79], complete multigraphs[80, 81], neutrosophic multigraphs [72, 82], soft multigraphs[83], and directed multigraphs [84] are known. An iterated multigraph uses iterated multisets as vertex objects, so vertices themselves can be multisets nested to depth n recursively.

Definition 2.2.1 (Finite multiset and iterated multiset)**.** Let X be a set. A *finite multiset* on X is a function

$$m : X \to \mathbb{N}_0$$

whose *support*

$$\mathrm{supp}(m) := \{x \in X \mid m(x) > 0\}$$

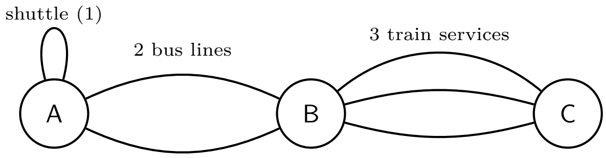


Figure 2.3: Public transport routes modeled as an undirected multigraph $G = (V, \mu)$ in Example 2.2.4: two parallel edges between A and B, three parallel edges between B and C, and a loop at A.

is finite. We write $\mathsf{M}(X)$ for the set of all finite multisets on X.

For $n \geq 0$, define the *n-fold iterated multiset sets* recursively by

$$\mathsf{M}^0(X) := X, \qquad \mathsf{M}^{n+1}(X) := \mathsf{M}\big(\mathsf{M}^n(X)\big).$$

An element of $\mathsf{M}^n(X)$ is called an *n-fold iterated multiset over X*.

Definition 2.2.2 (MultiGraph (undirected multigraph))**.** Let V be a finite set. Denote by

$$\binom{V}{2}^m := \big\{\{\{u, v\}\} \;\big|\; u, v \in V\big\}$$

the set of *unordered pairs with repetition* (i.e. 2-element multisets), so that $\{\{v, v\}\}$ represents a loop at v.

A *MultiGraph* (undirected multigraph) on V is a pair

$$G = (V, \mu),$$

where $\mu : \binom{V}{2}^m \to \mathbb{N}_0$ is an *edge-multiplicity function*. For $e \in \binom{V}{2}^m$, the value $\mu(e)$ is the number of parallel edges of type e. Equivalently, one may specify a finite multiset $E \in \mathsf{M}\Big(\binom{V}{2}^m\Big)$ and write $G = (V, E)$, where μ is the multiplicity function associated to E.

Remark 2.2.3 (Directed variant)**.** *[85] A* directed *multigraph can be defined similarly by a multiplicity map $\mu : V \times V \to \mathbb{N}_0$, where $\mu(u, v)$ counts the number of directed edges from u to v.*

Example 2.2.4 (Public transport routes as a MultiGraph)**.** Let

$$V = \{\mathsf{A}, \mathsf{B}, \mathsf{C}\}$$

be three stations. Suppose there are *two* distinct bus lines between A and B, *three* distinct train services between B and C, and one circular shuttle at A (a loop). Define the multiplicity map

$$\mu : \binom{V}{2}^m \to \mathbb{N}_0$$

by

$$\mu(\{\{\mathsf{A}, \mathsf{B}\}\}) = 2, \qquad \mu(\{\{\mathsf{B}, \mathsf{C}\}\}) = 3, \qquad \mu(\{\{\mathsf{A}, \mathsf{A}\}\}) = 1,$$

and $\mu(e) = 0$ for all other $e \in \binom{V}{2}^m$. Then $G = (V, \mu)$ is an undirected multigraph in the sense of Definition 2.2.2. For reference, an overview diagram is provided in Fig. 2.3.

Definition 2.2.5 (Iterated MultiGraph of order n)**.** Let X be a nonempty base set and let $n \geq 0$. Set $\mathsf{M}^n(X)$ as in Definition 2.2.1. An *Iterated MultiGraph of order n over X* is an undirected multigraph whose vertex objects are n-fold iterated multisets over X; concretely, it is a pair

$$G^{(n)} = (V^{(n)}, \mu^{(n)})$$

such that:

1. $V^{(n)} \in \mathsf{M}^n(X)$ is an n-fold iterated multiset, interpreted as a *vertex multiset*; let

$$\underline{V}^{(n)} := \operatorname{supp}(V^{(n)}) \subseteq \mathsf{M}^n(X)$$

be the underlying *set* of distinct vertices.

Table 2.3: A concise comparison of Graph, MultiGraph, and Iterated MultiGraph.

Concept	**Vertex domain**	**Edge structure**	**Main feature**
Graph	Ordinary vertices	Ordinary edges between vertex pairs	Represents simple pairwise adjacency without edge multiplicity.
MultiGraph	Ordinary vertices	Edges with multiplicity between the same vertex pair (and possibly loops)	Allows repeated pairwise connections while keeping the vertex set classical.
Iterated MultiGraph	Iterated multiset-based vertices	Ordinary multiset edges on those higher-level vertex objects	Extends multigraph structure by introducing recursive multiset organization on the vertex side.

2. $\mu^{(n)} : \binom{\underline{V}^{(n)}}{2}^m \to \mathbb{N}_0$ is an edge-multiplicity function on unordered pairs (with repetition) of distinct vertices.

Equivalently, one may specify an edge multiset $E^{(n)} \in \mathsf{M}\left(\binom{\underline{V}^{(n)}}{2}^m\right)$ and write $G^{(n)} = (V^{(n)}, E^{(n)})$.

Remark 2.2.6 (Order 0 recovers ordinary multigraphs)**.** *When $n = 0$, we have $\mathsf{M}^0(X) = X$, so an Iterated MultiGraph of order 0 is just a multigraph whose vertices lie in the base set X (up to the choice of vertex multiset versus vertex set).*

Example 2.2.7 (Iterated MultiGraph of order 1 (multiset-vertices))**.** Let the base set be

$$X = \{a, b, c\}.$$

Consider the following 1-fold iterated multiset (a multiset of elements of X) as the vertex multiset:

$$V^{(1)} = \{\{\ \{\{a\}\},\ \{\{a\}\},\ \{\{b\}\},\ \{\{c\}\}\ \}\} \in \mathsf{M}^1(X) = \mathsf{M}(X).$$

Hence the underlying set of distinct vertices is

$$\underline{V}^{(1)} = \mathrm{supp}(V^{(1)}) = \{\{\{a\}\},\ \{\{b\}\},\ \{\{c\}\}\} \subseteq \mathsf{M}(X),$$

where we view each vertex as a 1-multiset (e.g. $\{\{a\}\}$).

Define an edge-multiplicity function

$$\mu^{(1)} : \binom{\underline{V}^{(1)}}{2}^m \to \mathbb{N}_0$$

by

$$\mu^{(1)}(\{\{\{\{a\}\}, \{\{b\}\}\}\}) = 2, \qquad \mu^{(1)}(\{\{\{\{a\}\}, \{\{c\}\}\}\}) = 1,$$

and $\mu^{(1)}(e) = 0$ for all other e. Then

$$G^{(1)} = \left(V^{(1)}, \mu^{(1)}\right)$$

is an Iterated MultiGraph of order 1 over X in the sense of Definition 2.2.5. Intuitively, the vertices are *multiset-objects* built from X, and edges connect these multiset-vertices with possible parallel multiplicities.

A concise comparison of Graph, MultiGraph, and Iterated MultiGraph is given in Table 2.3.

2.3 h-model

An *h-model* is a structure $\langle S, \mathcal{H}, I\rangle$ consisting of a base set S, a finite collection $\mathcal{H}$ of hypergraphs on S, and an interpretation map I assigning each propositional atom a hypergraph in $\mathcal{H}$ [86].

Definition 2.3.1 (*h*-model)**.** [86] Let S be a nonempty set and write

$$\mathcal{P}^*(S) := \mathcal{P}(S) \setminus \{\emptyset\}.$$

A *(simple) hypergraph on S* is a finite family $H \subseteq \mathcal{P}^*(S)$.

Let At be a set of propositional atoms. An *h-model* is a triple

$$\mathfrak{M} = \langle S, \mathcal{H}, I\rangle$$

such that

1. $\mathcal{H} \subseteq \mathcal{P}(\mathcal{P}^*(S))$ is a finite collection of simple hypergraphs on S;
2. $I : \mathrm{At} \to \mathcal{H}$ assigns to each atom $p \in \mathrm{At}$ a hypergraph $I(p) \in \mathcal{H}$.

Example 2.3.2 (h-model for small-group access policies)**.** Let the base set of users be

$$S = \{\mathsf{Alice}, \mathsf{Bob}, \mathsf{Carol}\}.$$

Define two simple hypergraphs on S:

$$H_{\mathsf{pair}} = \{\{\mathsf{Alice}, \mathsf{Bob}\}, \{\mathsf{Alice}, \mathsf{Carol}\}, \{\mathsf{Bob}, \mathsf{Carol}\}\},$$

$$H_{\mathsf{team}} = \{\{\mathsf{Alice}, \mathsf{Bob}, \mathsf{Carol}\}, \{\mathsf{Alice}, \mathsf{Bob}\}\}.$$

Let

$$\mathcal{H} = \{H_{\mathsf{pair}}, H_{\mathsf{team}}\} \subseteq \mathcal{P}(\mathcal{P}^*(S)).$$

Take a set of propositional atoms

$$\mathrm{At} = \{p, q\},$$

and define the interpretation map $I : \mathrm{At} \to \mathcal{H}$ by

$$I(p) = H_{\mathsf{pair}}, \qquad I(q) = H_{\mathsf{team}}.$$

Then $\mathfrak{M} = \langle S, \mathcal{H}, I \rangle$ is an h-model in the sense of Definition 2.3.1. Intuitively, the atom p is interpreted as the hypergraph of all two-person approvals, while q is interpreted as a hypergraph of larger team approvals. An overview diagram of this example is provided in Fig. 2.4.

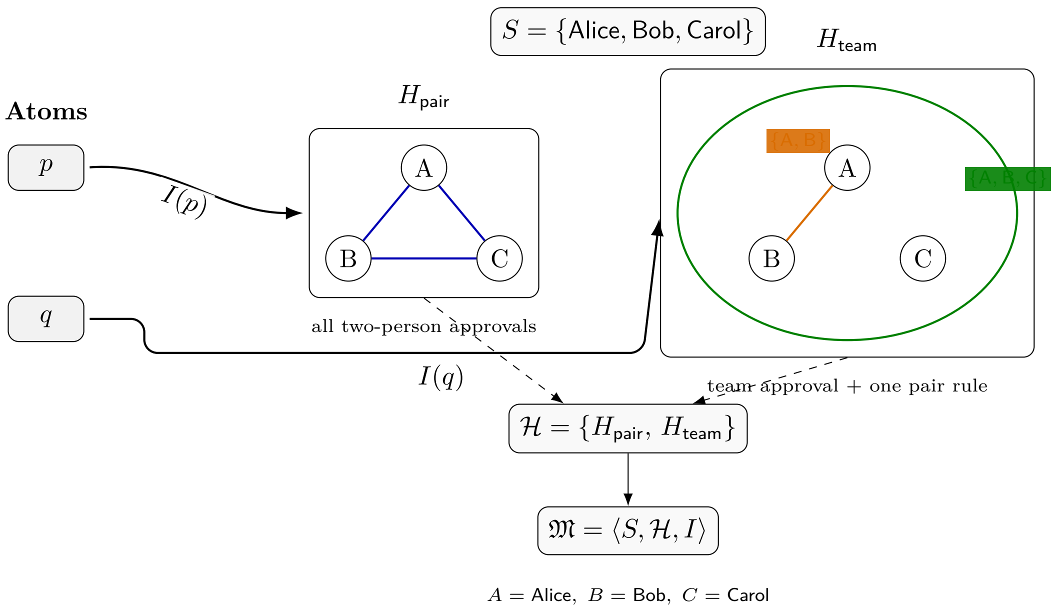


Figure 2.4: An illustration of the h-model in Example 2.3.2.

An *sh-model* may be regarded as a SuperHyperGraph-based analogue of an h-model: instead of assigning a hypergraph to each propositional atom, it assigns a finite n-SuperHyperGraph over a fixed base set.

Definition 2.3.3 ((Recall) Finite n-SuperHyperGraph over a base set)**.** Let S be a nonempty set and let $n \in \mathbb{N}_0$. Write

$$\mathcal{P}^0(S) := S, \qquad \mathcal{P}^{k+1}(S) := \mathcal{P}(\mathcal{P}^k(S)) \quad (k \geq 0),$$

and

$$\mathcal{P}^*(X) := \mathcal{P}(X) \setminus \{\varnothing\}.$$

A *finite n-SuperHyperGraph over S* is a triple

$$G = (V, E, \partial),$$

such that

1. $V \subseteq \mathcal{P}^n(S)$ is a finite set;
2. E is a finite set;
3. $$\partial : E \longrightarrow \mathcal{P}^*(V)$$
   is a map.

We denote by
$$\mathbf{SHG}_n(S)$$
the class of all finite n-SuperHyperGraphs over S.

Definition 2.3.4 (*sh*-model)**.** Let S be a nonempty set, let $n \in \mathbb{N}_0$, and let At be a set of propositional atoms. An *sh-model of level* n is a quadruple
$$\mathfrak{M}^{(n)} = \langle S, \mathbf{SH}, I, n \rangle$$
such that

1. $$\mathbf{SH} \subseteq \mathbf{SHG}_n(S)$$
   is a finite collection of finite n-SuperHyperGraphs over S;
2. $$I : \mathrm{At} \longrightarrow \mathbf{SH}$$
   is an interpretation map assigning to each atom $p \in \mathrm{At}$ a member $I(p) \in \mathbf{SH}$.

Remark 2.3.5. *If $n = 0$, then $\mathcal{P}^0(S) = S$, and each finite 0-SuperHyperGraph over S is simply a finite hypergraph written in incidence-map form. Hence an sh-model of level 0 reduces to a hypergraph-based model of the same general type as an h-model.*

Theorem 2.3.6 (Well-definedness of *sh*-models)**.** *Let S be a nonempty set, let $n \in \mathbb{N}_0$, and let* At *be a set of propositional atoms. Suppose that*
$$\mathbf{SH} \subseteq \mathbf{SHG}_n(S)$$
is a finite family of finite n-SuperHyperGraphs over S, and let
$$I : \mathrm{At} \to \mathbf{SH}$$
be any map. Then
$$\mathfrak{M}^{(n)} = \langle S, \mathbf{SH}, I, n \rangle$$
is a well-defined sh-model in the sense of Definition 2.3.4.

Proof. We verify the two clauses of Definition 2.3.4.

Since $n \in \mathbb{N}_0$ and $S \neq \varnothing$, the iterated powerset $\mathcal{P}^n(S)$ is well-defined by recursion. Hence the notion of a finite n-SuperHyperGraph over S is meaningful by Definition 2.3.3. Therefore $\mathbf{SHG}_n(S)$ is a well-defined class of objects, and the assumption
$$\mathbf{SH} \subseteq \mathbf{SHG}_n(S)$$
means precisely that every member of $\mathbf{SH}$ is a finite n-SuperHyperGraph over S. By hypothesis, $\mathbf{SH}$ is finite, so condition (1) of Definition 2.3.4 holds.

Next, the map
$$I : \mathrm{At} \to \mathbf{SH}$$
is assumed to be a function. Hence for each propositional atom $p \in \mathrm{At}$, there exists a unique value $I(p) \in \mathbf{SH}$. Thus condition (2) of Definition 2.3.4 also holds.

Therefore all constituents of
$$\mathfrak{M}^{(n)} = \langle S, \mathbf{SH}, I, n \rangle$$
are well-defined and satisfy the required properties. Consequently, $\mathfrak{M}^{(n)}$ is a well-defined *sh*-model. □

2.4 Chain-Free Subsets

A *k-chain-free subset* of a poset is a subset that contains no strictly increasing chain of length k; equivalently, it avoids k mutually comparable elements [87].

Definition 2.4.1 (k-chain)**.** [87] Let $P = (X, \leq)$ be a finite poset and let $k \in \mathbb{N}$. A *chain of length k* (a *k-chain*) is a k-tuple $(x_1, \dots, x_k)$ of elements of X such that

$$x_1 < x_2 < \cdots < x_k.$$

Definition 2.4.2 (k-chain-free subset and $\mathrm{Forb}_k(P)$)**.** [87] Let $P = (X, \leq)$ be a finite poset and let $k \geq 2$. A subset $A \subseteq X$ is *k-chain-free* if it contains no k-chain; i.e., there do not exist distinct $a_1, \dots, a_k \in A$ with

$$a_1 < a_2 < \cdots < a_k.$$

We denote by

$$\mathrm{Forb}_k(P) := \{\, A \subseteq X \mid A \text{ is } k\text{-chain-free and maximal w.r.t. inclusion} \,\}$$

the family of inclusion-maximal k-chain-free subsets.

Example 2.4.3 (2-chain-free subsets in a Boolean poset)**.** Let $P = (X, \subseteq)$ be the Boolean poset on

$$X = \mathcal{P}(\{1, 2, 3\})$$

ordered by inclusion. Take $k = 2$. Then a subset $A \subseteq X$ is 2-*chain-free* precisely when it contains no comparable pair (i.e., no $A_1, A_2 \in A$ with $A_1 \subset A_2$). For instance, the *middle layer*

$$A = \{\{1, 2\}, \{1, 3\}, \{2, 3\}\}$$

is 2-chain-free, because any two distinct 2-subsets of $\{1, 2, 3\}$ are incomparable. Moreover, A is maximal with respect to inclusion among 2-chain-free subsets of X, hence

$$A \in \mathrm{Forb}_2(P).$$

An overview diagram of this example is provided in Fig. 2.5.

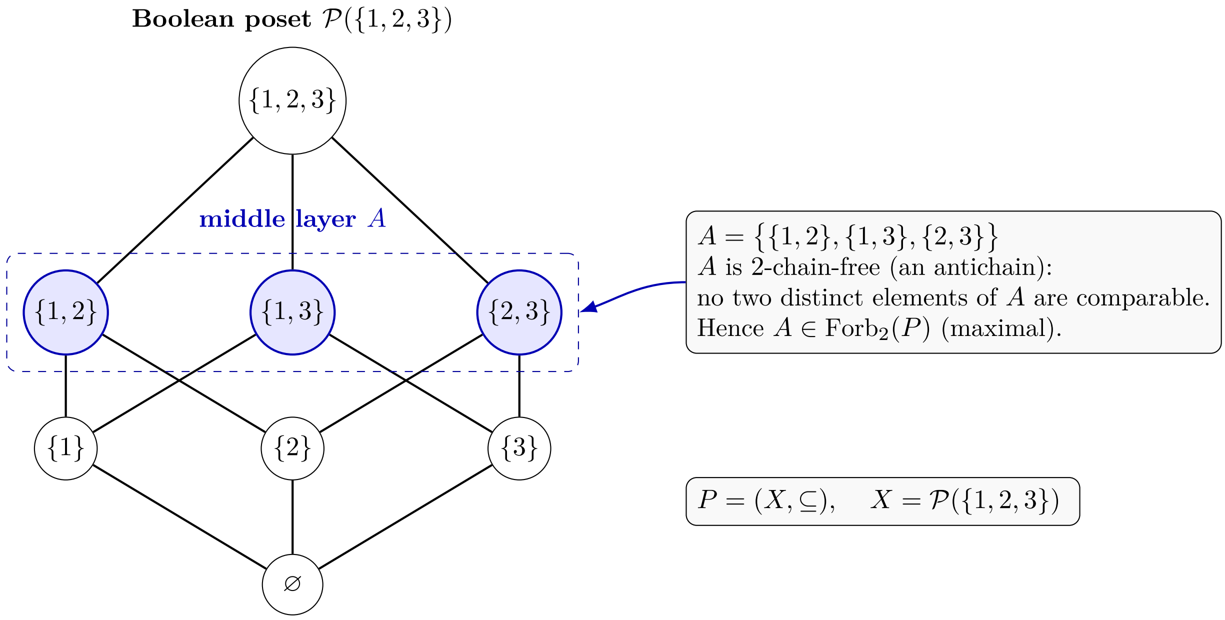


Figure 2.5: Hasse diagram of the Boolean poset on $\mathcal{P}(\{1, 2, 3\})$, highlighting the 2-chain-free middle layer A.

The iterated notion of a k-chain-free subset can also be defined as follows.

Definition 2.4.4 (Iterated k-chain-free subset (depth r))**.** Let $P = (X, \leq)$ be a finite poset and let $k \geq 2$. Define recursively a sequence of posets

$$P_k^{(r)} = \left(U_k^{(r)},\ \preceq^{(r)}\right) \qquad (r \in \mathbb{N}_0)$$

as follows:

1. **(Depth** 0**)** Set $U_k^{(0)} := X$ and $\preceq^{(0)} := \leq$.
2. **(Depth** $r+1$**)** Having defined $P_k^{(r)} = (U_k^{(r)}, \preceq^{(r)})$, let

$$U_k^{(r+1)} := \left\{ A \subseteq U_k^{(r)} \;\middle|\; A \text{ is } k\text{-chain-free in the poset } (U_k^{(r)}, \preceq^{(r)}) \right\},$$

and equip $U_k^{(r+1)}$ with the inclusion order

$$A \preceq^{(r+1)} B \quad :\Longleftrightarrow \quad A \subseteq B \qquad (A, B \in U_k^{(r+1)}).$$

An element of $U_k^{(r)}$ is called an *iterated k-chain-free subset of depth r* (over P). Equivalently, it is a "k-chain-free set of k-chain-free sets of $\cdots$ of k-chain-free sets" (r iterations), where the first iteration uses $\leq$ on X, and higher iterations use inclusion.

Remark 2.4.5 (Maximal (iterated) k-chain-free families)**.** *If one prefers inclusion*-maximal *objects at each depth, define*

$$\mathrm{Forb}_k^{(r)}(P) := \left\{ A \in U_k^{(r)} \;\middle|\; A \text{ is maximal in } (U_k^{(r)}, \subseteq) \right\} \qquad (r \geq 1),$$

so $\mathrm{Forb}_k^{(1)}(P)$ *coincides with the family of maximal k-chain-free subsets of P.*

Example 2.4.6 (An iterated 2-chain-free subset of depth 2)**.** Let $P = (X, \subseteq)$ again be the Boolean poset on $X = \mathcal{P}(\{1,2,3\})$, and let $k = 2$. As in Example 2.4.3, elements of $U_2^{(1)}$ are 2-chain-free families of subsets of $\{1,2,3\}$. Consider the following three maximal 2-chain-free families (each is an antichain in X):

$$A_1 = \{\{1\},\{2\},\{3\}\}, \qquad A_2 = \{\{1,2\},\{1,3\},\{2,3\}\}, \qquad A_3 = \{\emptyset, \{1,2,3\}\}.$$

Each A_i is 2-chain-free in $(X, \subseteq)$, so $A_1, A_2, A_3 \in U_2^{(1)}$. Now form a depth-2 object (a set of 2-chain-free sets) by

$$B = \{A_1, A_2\} \subseteq U_2^{(1)}.$$

Since $U_2^{(1)}$ is ordered by inclusion of families, and A_1 and A_2 are incomparable (neither is a subset of the other), the set B contains no 2-chain in the poset $(U_2^{(1)}, \subseteq)$. Hence B is 2-chain-free in $(U_2^{(1)}, \subseteq)$, and therefore

$$B \in U_2^{(2)}.$$

Thus B is an iterated 2-chain-free subset of depth 2 in the sense of Definition 2.4.4.

Theorem 2.4.7 (Well-definedness of iterated k-chain-free subsets)**.** *Let $P = (X, \leq)$ be a finite poset and let $k \geq 2$. Define $P_k^{(r)} = (U_k^{(r)}, \preceq^{(r)})$ recursively as in Definition 2.4.4. Then for every $r \in \mathbb{N}_0$:*

(i) *$U_k^{(r)}$ is a well-defined finite set, and in particular $U_k^{(r+1)} \subseteq \mathcal{P}(U_k^{(r)})$.*

(ii) *$\preceq^{(r)}$ is a well-defined partial order on $U_k^{(r)}$; hence $P_k^{(r)}$ is a finite poset.*

Consequently, the phrase "iterated k-chain-free subset of depth r" is well-defined.

Proof. We argue by induction on r.
Base case $r = 0$. By definition, $U_k^{(0)} = X$ and $\preceq^{(0)} = \leq$. Since $P = (X, \leq)$ is assumed to be a finite poset, $U_k^{(0)}$ is a finite set and $\preceq^{(0)}$ is a partial order on it. Thus $P_k^{(0)}$ is well-defined as a finite poset.
Inductive step. Assume that for some $r \in \mathbb{N}_0$ the structure $P_k^{(r)} = (U_k^{(r)}, \preceq^{(r)})$ is a well-defined finite poset. We must show that $P_k^{(r+1)} = (U_k^{(r+1)}, \preceq^{(r+1)})$ is a well-defined finite poset.
(i) $U_k^{(r+1)}$ is well-defined and finite. Because $(U_k^{(r)}, \preceq^{(r)})$ is a poset by the induction hypothesis, the notion "A is k-chain-free in $(U_k^{(r)}, \preceq^{(r)})$" is meaningful; explicitly, $A \subseteq U_k^{(r)}$ is k-chain-free if there do not exist elements $x_1, \ldots, x_k \in A$ such that

$$x_1 \prec^{(r)} x_2 \prec^{(r)} \cdots \prec^{(r)} x_k.$$

Hence the specification

$$U_k^{(r+1)} = \{ A \subseteq U_k^{(r)} \mid A \text{ is } k\text{-chain-free in } (U_k^{(r)}, \preceq^{(r)}) \}$$

defines a bona fide subset of the power set $\mathcal{P}(U_k^{(r)})$. Moreover, since $U_k^{(r)}$ is finite, $\mathcal{P}(U_k^{(r)})$ is finite, and therefore $U_k^{(r+1)}$ is also finite.

(ii) $\preceq^{(r+1)}$ *is a partial order.* By definition, for $A, B \in U_k^{(r+1)}$ we set

$$A \preceq^{(r+1)} B \iff A \subseteq B.$$

Since inclusion $\subseteq$ is reflexive, antisymmetric, and transitive on *all* subsets of $U_k^{(r)}$, its restriction to the subcollection $U_k^{(r+1)} \subseteq \mathcal{P}(U_k^{(r)})$ is again reflexive, antisymmetric, and transitive. Thus $\preceq^{(r+1)}$ is a well-defined partial order on $U_k^{(r+1)}$.

Combining (i) and (ii), $P_k^{(r+1)}$ is a well-defined finite poset. This completes the induction and proves the theorem. □

2.5 Power Set Graph

A *Power Set Graph* has vertices given by the nonempty proper subsets of a set, with edges between comparable subsets by inclusion [88]. An *Iterated Power Set Graph* repeatedly applies the nontrivial powerset construction; at each depth, vertices are nested subsets, adjacent when comparable by inclusion.

Definition 2.5.1 (Power set graph)**.** [88] Let A be a finite set with $|A| \geq 2$. The *power set graph* of A, denoted $\Gamma(\mathcal{P}(A))$, is the simple undirected graph

$$\Gamma(\mathcal{P}(A)) = (V, E),$$

where

$$V = \mathcal{P}(A) \setminus \{\emptyset, A\}, \qquad E = \big\{\{X, Y\} \subseteq V \;\big|\; X \subset Y \text{ or } Y \subset X\big\}.$$

Equivalently, vertices are the nonempty proper subsets of A, and two vertices are adjacent iff one subset is contained in the other.

Example 2.5.2 (Power set graph on a 3-element set)**.** Let

$$A = \{1, 2, 3\}.$$

Then the vertex set of the power set graph $\Gamma(\mathcal{P}(A))$ is

$$V = \mathcal{P}(A) \setminus \{\emptyset, A\} = \big\{\{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}\big\}.$$

Two vertices are adjacent iff one is a subset of the other. For instance,

$$\{1\} \sim \{1, 2\}, \qquad \{1\} \sim \{1, 3\}, \qquad \{2\} \sim \{1, 2\},$$

while $\{1, 2\} \not\sim \{1, 3\}$ because neither contains the other. Hence $\Gamma(\mathcal{P}(A))$ is the comparability graph of the inclusion poset on nonempty proper subsets of A, as in Definition 2.5.1.

Definition 2.5.3 (Iterated Power Set Graph)**.** Let A be a finite set with $|A| \geq 2$, and write

$$\mathcal{P}^*(S) := \mathcal{P}(S) \setminus \{\emptyset, S\}$$

for the family of *nonempty proper* subsets of a finite set S.

Define recursively the *iterated nontrivial powerset universes* $(U_r(A))_{r \geq 0}$ by

$$U_0(A) := A, \qquad U_{r+1}(A) := \mathcal{P}^*\big(U_r(A)\big) \quad (r \geq 0).$$

For each $r \geq 1$, the *Iterated Power Set Graph of depth* r on A is the simple undirected graph

$$\Gamma_r(A) = (V_r, E_r),$$

where

$$V_r := U_r(A), \qquad E_r := \Big\{\{X, Y\} \subseteq V_r \;\Big|\; X \subset Y \text{ or } Y \subset X\Big\}.$$

Thus, vertices at depth r are nonempty proper subsets of $U_{r-1}(A)$, and two vertices are adjacent if and only if they are comparable by inclusion.

In particular, $\Gamma_1(A)$ is the usual power set graph on A. (Optionally, one may set $\Gamma_0(A)$ to be the edgeless graph on vertex set A.)

Example 2.5.4 (Iterated power set graph of depth 2)**.** Let

$$A = \{1, 2, 3\}.$$

At depth 1,

$$U_1(A) = \mathcal{P}^*(A) = \{\{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}\},$$

so $\Gamma_1(A)$ is the power set graph from Example 2.5.2. At depth 2, the vertex universe is

$$V_2 = U_2(A) = \mathcal{P}^*(U_1(A)),$$

whose elements are the nonempty proper *subsets of* $U_1(A)$. For example, the following are vertices of $\Gamma_2(A)$:

$$X = \{\{1\}, \{1, 2\}\} \in U_2(A), \qquad Y = \{\{1\}, \{1, 2\}, \{2\}\} \in U_2(A),$$

and they are adjacent in $\Gamma_2(A)$ because $X \subset Y$. By contrast,

$$Z = \{\{3\}, \{2, 3\}\} \in U_2(A)$$

is not adjacent to Y because neither $Z \subset Y$ nor $Y \subset Z$ holds. Thus $\Gamma_2(A)$ is a graph whose vertices are *sets of nontrivial subsets of* A, adjacent when comparable by inclusion, exactly as in Definition 2.5.3. An overview diagram of this example is provided in Fig. 2.6.

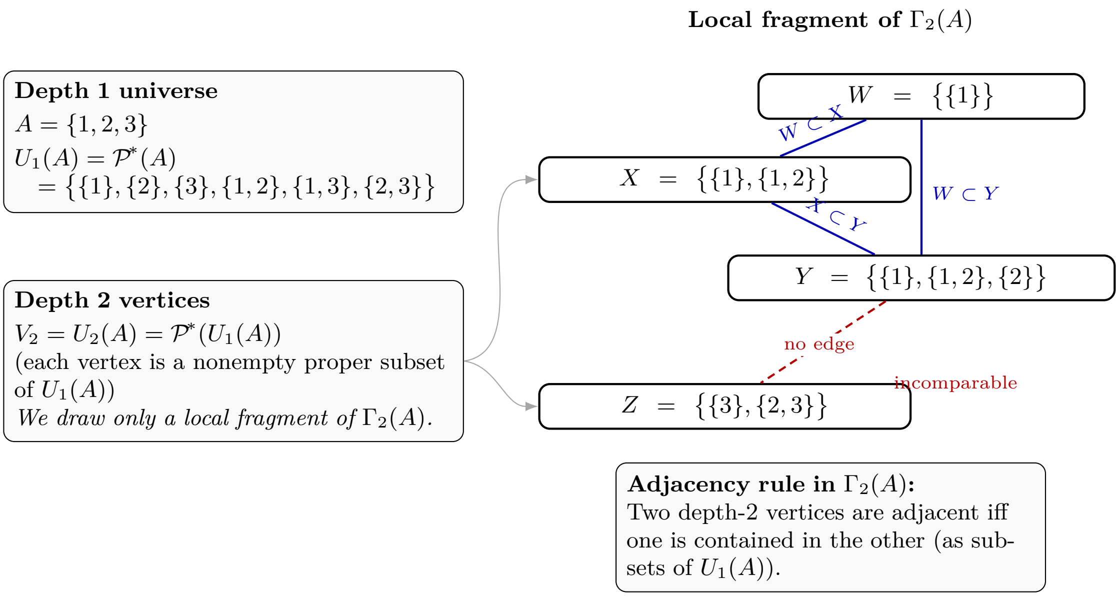


Figure 2.6: Illustration of an iterated power set graph of depth 2 for $A = \{1, 2, 3\}$. The figure shows a local induced fragment of $\Gamma_2(A)$ with example vertices X, Y, Z, W.

2.6 Johnson Graph

A *Johnson graph* has vertices given by the w-subsets of $[n]$, with edges joining pairs that differ by one element [89, 90, 91]. Related notions of the Johnson graph are also known, such as the generalized Johnson graph[92, 93].

Definition 2.6.1 (Johnson graph)**.** [94, 91] Let $n, w \in \mathbb{N}$ with $0 < w < n$, and write $[n] := \{1, 2, \ldots, n\}$. The *Johnson graph* $J(n, w)$ is the simple graph

$$J(n, w) = (V, E),$$

where

$$V = \{\, X \subseteq [n] \mid |X| = w \,\}, \qquad E = \{\{X, Y\} \subseteq V \;\mid\; |X \cap Y| = w - 1\}.$$

Thus two w-subsets are adjacent exactly when they differ by one element.

Example 2.6.2 (The Johnson graph $J(4,2)$)**.** Let $n = 4$ and $w = 2$. Then the vertex set of the Johnson graph $J(4,2)$ consists of all 2-subsets of $[4] = \{1,2,3,4\}$:

$$V = \{\{1,2\},\{1,3\},\{1,4\},\{2,3\},\{2,4\},\{3,4\}\}.$$

Two vertices are adjacent if they differ by exactly one element (equivalently, their intersection has size 1). For instance,

$$\{1,2\} \sim \{1,3\}, \qquad \{1,2\} \sim \{2,4\}, \qquad \{2,3\} \sim \{3,4\},$$

since

$$|\{1,2\} \cap \{1,3\}| = 1, \quad |\{1,2\} \cap \{2,4\}| = 1, \quad |\{2,3\} \cap \{3,4\}| = 1.$$

By contrast, $\{1,2\} \not\sim \{3,4\}$ because $\{1,2\} \cap \{3,4\} = \emptyset$. Hence $J(4,2)$ is the graph on 2-subsets of [4] with edges joining pairs that differ in exactly one element, as in Definition 2.6.1.

2.7 Kneser Graph

A *Kneser graph* has vertices given by the k-subsets of $[n]$, with edges joining pairs that are disjoint [95, 96, 97, 98]. Related concepts are also known, such as Kneser hypergraphs [99, 100], bipartite kneser graphs[101, 102], stable kneser graphs[103, 104], and generalized Kneser graphs [105, 106].

Definition 2.7.1 (Kneser graph)**.** [95, 96] Let $n, k \in \mathbb{N}$ with $n \geq 2k$, and write $[n] := \{1,2,\ldots,n\}$. The *Kneser graph* $\mathrm{KG}_{n,k}$ is the simple graph

$$\mathrm{KG}_{n,k} = (V,E),$$

where

$$V = \{\, X \subseteq [n] \mid |X| = k \,\}, \qquad E = \{\{X,Y\} \subseteq V \;\mid\; X \cap Y = \emptyset\}.$$

Thus two k-subsets are adjacent exactly when they are disjoint.

Example 2.7.2 (The Kneser graph $\mathrm{KG}_{5,2}$ (the Petersen graph))**.** Let $n = 5$ and $k = 2$. Then the vertex set of the Kneser graph $\mathrm{KG}_{5,2}$ consists of all 2-subsets of $[5] = \{1,2,3,4,5\}$:

$$V = \{\{1,2\},\{1,3\},\{1,4\},\{1,5\},\{2,3\},\{2,4\},\{2,5\},\{3,4\},\{3,5\},\{4,5\}\}.$$

Two vertices are adjacent if and only if the corresponding 2-subsets are disjoint. For example,

$$\{1,2\} \sim \{3,4\}, \qquad \{1,5\} \sim \{2,3\}, \qquad \{2,4\} \sim \{1,3\},$$

since

$$\{1,2\} \cap \{3,4\} = \emptyset, \quad \{1,5\} \cap \{2,3\} = \emptyset, \quad \{2,4\} \cap \{1,3\} = \emptyset.$$

By contrast, $\{1,2\} \not\sim \{2,5\}$ because $\{1,2\} \cap \{2,5\} = \{2\} \neq \emptyset$. Hence $\mathrm{KG}_{5,2}$ is the graph on 2-subsets of [5] with edges joining disjoint pairs, as in Definition 2.7.1. (It is well known that $\mathrm{KG}_{5,2}$ is isomorphic to the Petersen graph.)

2.8 Meta-Graph and Iterated Meta-Graph

A meta-graph has graphs as vertices; edges represent labeled relations between those graphs, satisfying relation-defined incidence constraints (cf.[107, 108, 109, 110, 111]). A meta-graph is also called a graph of graphs. An iterated meta-graph repeats the construction: vertices are meta-graphs of lower depth, with lifted relations linking them recursively [107].

Definition 2.8.1 (Meta-Graph (Metagraph; graph of graphs))**.** [107] Fix a nonempty universe $\mathcal{G}$ of finite graphs (undirected and loopless by default), and fix a nonempty family $\mathcal{R}$ of binary relations on $\mathcal{G}$, i.e.

$$\mathcal{R} \subseteq \mathcal{P}(\mathcal{G} \times \mathcal{G}).$$

A *Meta-Graph* (or *metagraph*) over $(\mathcal{G}, \mathcal{R})$ is a directed, $\mathcal{R}$-labeled multigraph

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

such that

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

and the following *incidence constraint* holds:

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

Elements of V are called *meta-vertices* (each meta-vertex is itself a graph in $\mathcal{G}$), and each $e \in E$ is a *meta-edge* labeled by the relation $\lambda(e)$. If $\mathcal{R} = \{R\}$ is a singleton, the label map may be omitted; if each $R \in \mathcal{R}$ is symmetric, one may view M as an undirected labeled multigraph.

Example 2.8.2 (A Meta-Graph of module-dependency graphs linked by a compatibility relation)**.** Let $\mathcal{G}$ be a universe consisting of three finite (simple, undirected) graphs that represent dependency structures of three software modules:

$$G_A = (V_A, E_A), \qquad G_B = (V_B, E_B), \qquad G_C = (V_C, E_C) \ \in \ \mathcal{G}.$$

For concreteness, take

$$\begin{aligned} V_A = \{a_1, a_2, a_3\},\ E_A = \big\{\{a_1, a_2\}, \{a_2, a_3\}\big\}, \\ V_B = \{b_1, b_2, b_3\},\ E_B = \big\{\{b_1, b_2\}, \{b_1, b_3\}\big\}, \\ V_C = \{c_1, c_2\},\ E_C = \big\{\{c_1, c_2\}\big\}. \end{aligned}$$

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

$$\begin{aligned} R_{\text{api}} &:= \big\{(G_i, G_j) \in \mathcal{G} \times \mathcal{G} \ \big|\ G_i \text{ and } G_j \text{ have mutually compatible APIs}\big\}, \\ R_{\text{data}} &:= \big\{(G_i, G_j) \in \mathcal{G} \times \mathcal{G} \ \big|\ G_i \text{ can safely consume data produced by } G_j\big\}. \end{aligned}$$

Let $\mathcal{R} = \{R_{\text{api}}, R_{\text{data}}\} \subseteq \mathcal{P}(\mathcal{G} \times \mathcal{G})$.

Now define a directed $\mathcal{R}$-labeled multigraph

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

by taking

$$V = \{G_A, G_B, G_C\} \subseteq \mathcal{G}, \qquad E = \{e_1, e_2\},$$

and setting

$$\begin{aligned} s(e_1) = G_A, \quad t(e_1) = G_B, \quad \lambda(e_1) = R_{\text{api}}, \\ s(e_2) = G_B, \quad t(e_2) = G_C, \quad \lambda(e_2) = R_{\text{data}}. \end{aligned}$$

Assume that $(G_A, G_B) \in R_{\text{api}}$ and $(G_B, G_C) \in R_{\text{data}}$ hold, i.e. the API-compatibility and data-consumption conditions are satisfied for these pairs. Then the incidence constraint

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

holds, and hence M is a Meta-Graph over $(\mathcal{G}, \mathcal{R})$ in the sense of Definition 2.8.1. Intuitively, M is a "graph of graphs" that records how entire dependency graphs relate to each other under different semantic relations. An overview diagram of this example is provided in Fig. 2.7.

Definition 2.8.3 (Iterated Meta-Graph (depth t))**.** [107] Fix $(\mathcal{G}, \mathcal{R})$ as in Definition 2.8.1. Define recursively, for $t \in \mathbb{N}_0$, a universe of *level-t objects* $\mathcal{G}^{(t)}$ and a family of *level-t relations* $\mathcal{R}^{(t)}$ as follows:

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

Assume $\mathcal{G}^{(t)}$ and $\mathcal{R}^{(t)}$ are defined. Let $\mathcal{G}^{(t+1)}$ be the class of all finite metagraphs over $(\mathcal{G}^{(t)}, \mathcal{R}^{(t)})$ (i.e. all tuples $M = (V(M), E(M), s_M, t_M, \lambda_M)$ satisfying Definition 2.8.1 with $\mathcal{G}, \mathcal{R}$ replaced by $\mathcal{G}^{(t)}, \mathcal{R}^{(t)}$).

For each relation $R \in \mathcal{R}^{(t)}$, define its *lift* $R^{\uparrow} \subseteq \mathcal{G}^{(t+1)} \times \mathcal{G}^{(t+1)}$ by

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

Set

$$\mathcal{R}^{(t+1)} := \{\, R^{\uparrow} \mid R \in \mathcal{R}^{(t)} \,\}.$$

An *Iterated Meta-Graph of depth* t is then a metagraph

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

over $\big(\mathcal{G}^{(t)}, \mathcal{R}^{(t)}\big)$, i.e.

$$V^{(t)} \subseteq \mathcal{G}^{(t)}, \qquad \lambda^{(t)} : E^{(t)} \to \mathcal{R}^{(t)}, \qquad \forall e \in E^{(t)} :\ \big(s^{(t)}(e), t^{(t)}(e)\big) \in \lambda^{(t)}(e).$$

In particular, depth 0 iterated meta-graphs are ordinary metagraphs, and depth $t \geq 1$ iterated meta-graphs have vertices that are themselves metagraphs, recursively, up to t levels.

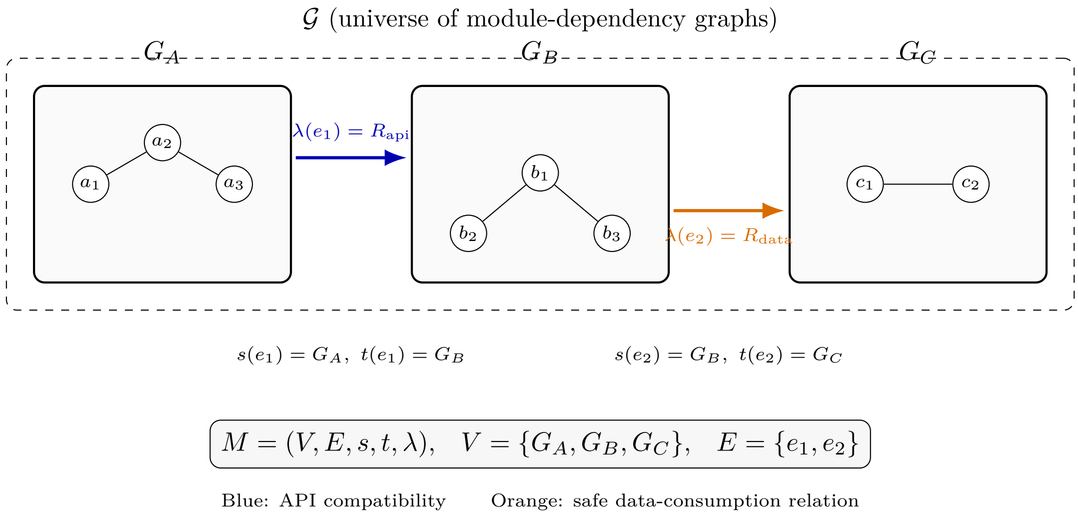


Figure 2.7: An illustration of the Meta-Graph in Example 2.8.2: each meta-vertex is itself a dependency graph, and meta-edges encode semantic relations between them.

Example 2.8.4 (A depth-1 Iterated Meta-Graph (a metagraph of metagraphs))**.** Continue with the universe $(\mathcal{G}, \mathcal{R})$ from Example 2.8.2. A level-1 object is a finite metagraph over $(\mathcal{G}, \mathcal{R})$.

Define two metagraphs $M_1, M_2 \in \mathcal{G}^{(1)}$ as follows. Let M_1 be the metagraph from Example 2.8.2, i.e.

$$M_1 = (\{G_A, G_B, G_C\},\ \{e_1, e_2\},\ s_1, t_1, \lambda_1),$$

with $\lambda_1(e_1) = R_{\text{api}}$ and $\lambda_1(e_2) = R_{\text{data}}$.

Let M_2 be another metagraph on the same vertex set

$$V(M_2) = \{G_A, G_B, G_C\}$$

with a single meta-edge f defined by

$$s_2(f) = G_A, \quad t_2(f) = G_C, \quad \lambda_2(f) = R_{\text{api}},$$

assuming $(G_A, G_C) \in R_{\text{api}}$ (module A is API-compatible with module C).

Lifted relations. By Definition 2.8.3, each relation $R \in \mathcal{R}^{(0)} = \mathcal{R}$ induces a lifted relation $R^{\uparrow} \in \mathcal{R}^{(1)}$ on $\mathcal{G}^{(1)}$:

$$(M_i, M_j) \in R^{\uparrow} \iff \exists x \in V(M_i), \exists y \in V(M_j) \text{ with } (x, y) \in R.$$

In particular, since M_1 contains the pair $(G_A, G_B) \in R_{\text{api}}$ and M_2 contains the pair $(G_A, G_C) \in R_{\text{api}}$, we have

$$(M_1, M_2) \in R^{\uparrow}_{\text{api}}.$$

Now define a level-1 metagraph (an iterated metagraph of depth 1)

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

by

$$V^{(1)} = \{M_1, M_2\} \subseteq \mathcal{G}^{(1)}, \qquad E^{(1)} = \{g\},$$
$$s^{(1)}(g) = M_1, \quad t^{(1)}(g) = M_2, \quad \lambda^{(1)}(g) = R^{\uparrow}_{\text{api}}.$$

Because $(M_1, M_2) \in R^{\uparrow}_{\text{api}}$, the incidence constraint holds:

$$(s^{(1)}(g), t^{(1)}(g)) \in \lambda^{(1)}(g).$$

Therefore $M^{(1)}$ is an Iterated Meta-Graph of depth 1 in the sense of Definition 2.8.3. Intuitively, it is a "metagraph of metagraphs" that relates two meta-level system descriptions whenever they contain underlying module pairs connected by R_{api}. An overview diagram of this example is provided in Fig. 2.8.

A concise comparison of Graph, Meta-Graph, and Iterated Meta-Graph is given in Table 2.4.

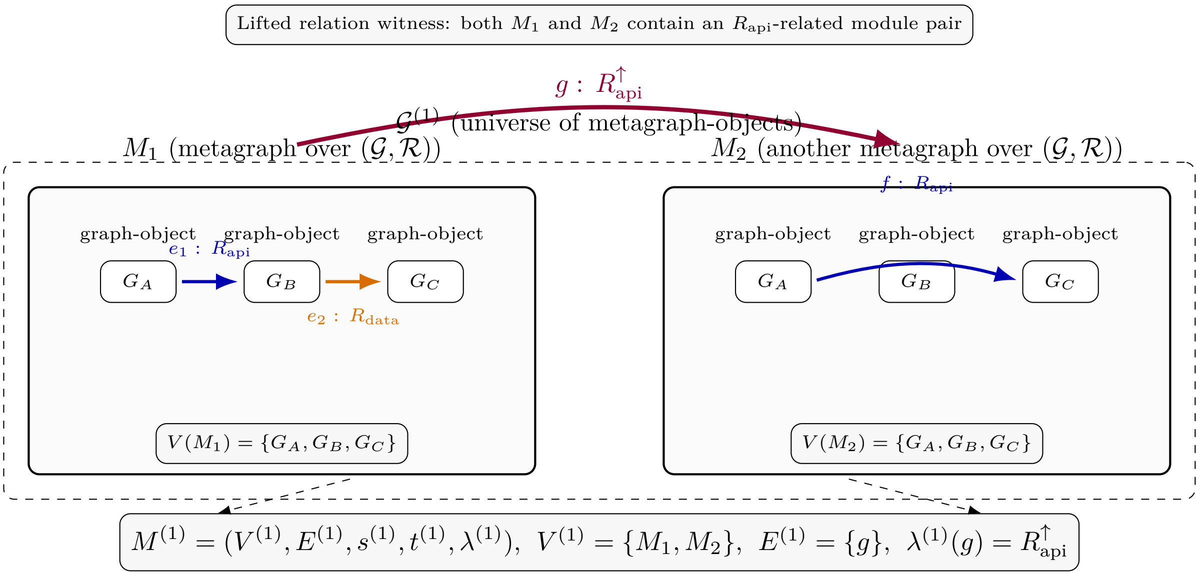


Figure 2.8: A depth-1 Iterated Meta-Graph: the vertices M_1, M_2 are themselves metagraphs, and the top-level edge g is labeled by the lifted relation $R^{\uparrow}_{\text{api}}$.

Table 2.4: A concise comparison of Graph, Meta-Graph, and Iterated Meta-Graph.

Concept	**Vertex domain**	**Edge meaning**	**Main feature**
Graph	Ordinary vertices	Ordinary adjacency between vertices	The classical pairwise network model.
Meta-Graph	Graphs	Labeled relations between graphs	A graph of graphs, where each vertex is itself a graph-object.
Iterated Meta-Graph	Meta-graphs of lower depth	Lifted relations between meta-graphs	A recursive graph-of-graphs-of-graphs construction across multiple meta-levels.

2.9 Meta-HyperGraph and Meta-SuperHyperGraph

A meta-hypergraph is a hypergraph whose vertices are objects; each hyperedge relates finite vertex sets via labeled relations (cf. [5, 112]). It can also be described as a *hypergraph of hypergraphs.* A meta-superhypergraph has vertices that are superhypergraphs; hyperedges relate finite collections of them, constrained by relation labels [5]. It can also be described as a *superhypergraph of superhypergraphs.*

Definition 2.9.1 (MetaHyperGraph over $(U, \mathcal{R})$)**.** [5] Let U be a nonempty universe of objects and let

$$\mathcal{R} \subseteq \mathcal{P}\big(\mathcal{P}_{\text{fin}}(U) \times \mathcal{P}_{\text{fin}}(U)\big)$$

be a nonempty family of *admissible set–relations.* A *MetaHyperGraph* over $(U, \mathcal{R})$ is a labelled directed hypergraph

$$M = (V, E, T, Hd, \lambda)$$

such that

$$V \subseteq U, \qquad T, Hd : E \to \mathcal{P}_{\text{fin}}(V), \qquad \lambda : E \to \mathcal{R},$$

and the *incidence constraint* holds:

$$\forall e \in E : \ (T(e), Hd(e)) \in \lambda(e).$$

The elements of V are called *meta-vertices.* If U is chosen to be a universe of (finite) hypergraphs, then M may be read as a *hypergraph of hypergraphs.*

Example 2.9.2 (A MetaHyperGraph of hypergraphs linked by an overlap relation)**.** Let the universe U be a collection of finite hypergraphs describing co-purchase groups in three cities:

$$U = \{H_T, H_O, H_N\},$$

where each $H_\bullet = (V_\bullet, E_\bullet)$ is a finite hypergraph on a product set $V_\bullet$. Define an admissible set–relation $R_{\text{ov}} \in \mathcal{P}\big(\mathcal{P}_{\text{fin}}(U) \times \mathcal{P}_{\text{fin}}(U)\big)$ by

$$(A, B) \in R_{\text{ov}} \quad :\Longleftrightarrow \quad \exists\, H \in A,\ \exists\, H' \in B \text{ such that } \frac{|V(H) \cap V(H')|}{|V(H) \cup V(H')|} \ \geq\ \theta,$$

for a fixed threshold $\theta \in (0, 1)$, where $V(H)$ denotes the vertex set of a hypergraph H. (Thus R_{ov} certifies that the two finite families contain at least one pair of hypergraphs whose vertex sets have sufficiently large Jaccard overlap.) Let

$$\mathcal{R} = \{R_{\text{ov}}\}.$$

Now form a labelled directed hypergraph

$$M = (V, E, T, Hd, \lambda)$$

as follows:

$$V = \{H_T, H_O, H_N\} \subseteq U, \qquad E = \{e_1, e_2\}.$$

Define tail and head maps $T, Hd : E \to \mathcal{P}_{\text{fin}}(V)$ by

$$T(e_1) = \{H_T, H_O\}, \quad Hd(e_1) = \{H_N\}, \qquad T(e_2) = \{H_O\}, \quad Hd(e_2) = \{H_T, H_N\},$$

and define the label map $\lambda : E \to \mathcal{R}$ by

$$\lambda(e_1) = R_{\text{ov}}, \qquad \lambda(e_2) = R_{\text{ov}}.$$

Assume that the chosen threshold θ is such that

$$\big(T(e_i), Hd(e_i)\big) \in R_{\text{ov}} \qquad (i = 1, 2),$$

i.e., each tail–head pair satisfies the overlap criterion. Then the incidence constraint

$$\forall e \in E :\ (T(e), Hd(e)) \in \lambda(e)$$

holds, and therefore M is a MetaHyperGraph over $(U, \mathcal{R})$ in the sense of Definition 2.9.1. Intuitively, M is a "hypergraph of hypergraphs" whose meta-hyperedges assert that certain *families* of city-level hypergraphs overlap strongly in their product catalogs. An overview diagram of this example is provided in Fig. 2.9.

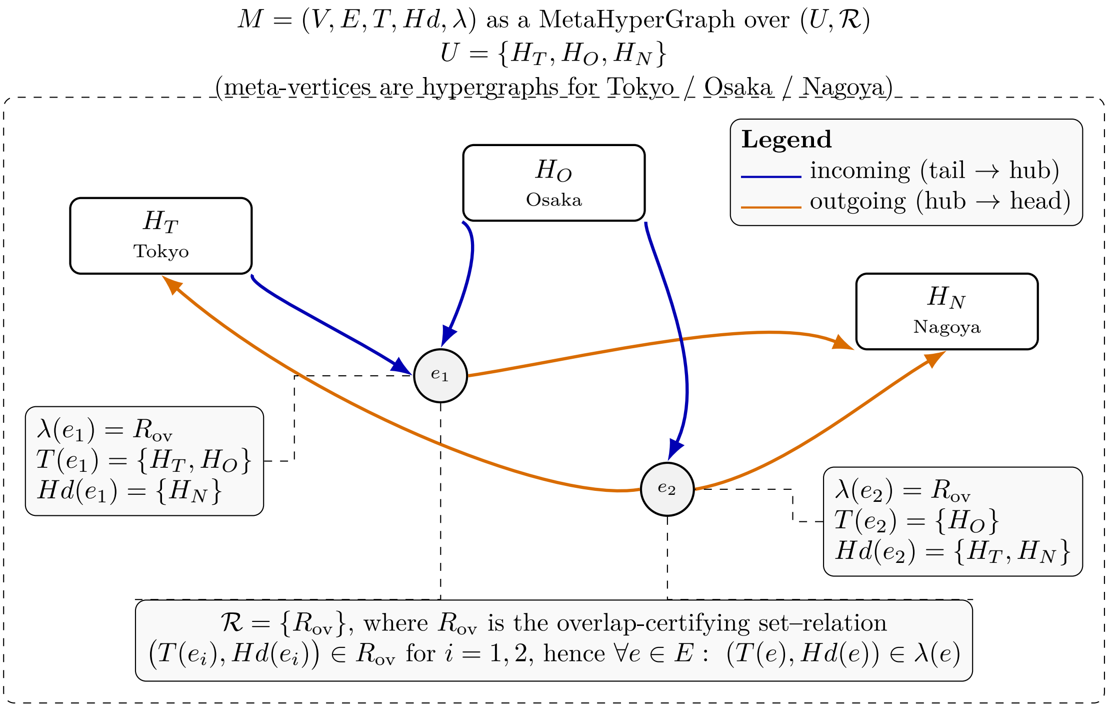


Figure 2.9: An illustration of the MetaHyperGraph in Example 2.9.2.

Definition 2.9.3 (MetaSuperHyperGraph over $(\mathsf{S}_n, \mathcal{R})$)**.** [5] Fix $n \in \mathbb{N}_0$ and let S_n denote the class of all finite directed n-SuperHyperGraphs (over arbitrary base sets). Let

$$\mathcal{R} \subseteq \mathcal{P}\big(\mathcal{P}_{\mathrm{fin}}(\mathsf{S}_n) \times \mathcal{P}_{\mathrm{fin}}(\mathsf{S}_n)\big)$$

be a nonempty family of admissible set–relations on S_n. A *MetaSuperHyperGraph* (abbrev. *MSHG*) over $(\mathsf{S}_n, \mathcal{R})$ is a labelled directed hypergraph

$$M = (V, E, T, Hd, \lambda)$$

such that

$$V \subseteq \mathsf{S}_n, \qquad T, Hd : E \to \mathcal{P}_{\mathrm{fin}}(V), \qquad \lambda : E \to \mathcal{R},$$

and the incidence constraint holds:

$$\forall e \in E: \ (T(e), Hd(e)) \in \lambda(e).$$

Thus vertices of M are (finite) n-SuperHyperGraphs, and each meta-hyperedge relates finite families of such vertices, certified by its label $\lambda(e) \in \mathcal{R}$.

Example 2.9.4 (A MetaSuperHyperGraph relating 1-SuperHyperGraphs by a refinement relation)**.** Fix $n = 1$. Let $V_0 = \{1, 2, 3, 4\}$ be a base set of atomic items. Consider three finite directed 1-SuperHyperGraphs (here written with undirected incidence for simplicity)

$$G_1 = (V_1^{(1)}, E_1^{(1)}, \partial_1), \qquad G_2 = (V_2^{(1)}, E_2^{(1)}, \partial_2), \qquad G_3 = (V_3^{(1)}, E_3^{(1)}, \partial_3),$$

over (possibly) different supervertex sets, where each $V_i^{(1)} \subseteq \mathcal{P}(V_0)$ is a finite set of 1-supervertices.

Define an admissible set–relation $R_{\mathrm{ref}} \in \mathcal{P}\big(\mathcal{P}_{\mathrm{fin}}(\mathsf{S}_1) \times \mathcal{P}_{\mathrm{fin}}(\mathsf{S}_1)\big)$ by

$$(A, B) \in R_{\mathrm{ref}} \quad :\Longleftrightarrow \quad \forall G \in A \ \exists G' \in B \text{ such that } G' \text{ is a supervertex-refinement of } G,$$

where "G' is a supervertex-refinement of G" means: for every supervertex $v' \in V(G')$ there exists $v \in V(G)$ with $v' \subseteq v$ (as subsets of V_0), and for every superedge $e' \in E(G')$ there exists $e \in E(G)$ such that

$$\bigcup \partial_{G'}(e') \ \subseteq \ \bigcup \partial_G(e) \quad \text{(as subsets of } V_0\text{)}.$$

Let

$$\mathcal{R} = \{R_{\mathrm{ref}}\}.$$

Now define a labelled directed hypergraph

$$M = (V, E, T, Hd, \lambda)$$

by taking

$$V = \{G_1, G_2, G_3\} \subseteq \mathsf{S}_1, \qquad E = \{e\}.$$

Set

$$T(e) = \{G_1\}, \qquad Hd(e) = \{G_2, G_3\}, \qquad \lambda(e) = R_{\mathrm{ref}}.$$

Assume that G_2 or G_3 (or both) refines G_1 in the above sense, so that

$$\big(T(e), Hd(e)\big) \in R_{\mathrm{ref}}.$$

Then the incidence constraint holds, hence M is a MetaSuperHyperGraph over $(\mathsf{S}_1, \mathcal{R})$ in the sense of Definition 2.9.3. Intuitively, the single meta-hyperedge asserts that the family $\{G_2, G_3\}$ contains refinements of G_1.

2.10 Nested HyperGraph and Nested SuperHyperGraph

A nested hypergraph allows hyperedges to contain other hyperedges as elements, with ranks enforcing well-founded nesting without cycles [113]. A nested superhypergraph uses supervertices from iterated powersets; superhyperedges may contain other superhyperedges, ranked to avoid cycles[113].

Definition 2.10.1 (Nested HyperGraph)**.** [113] Let V be a finite nonempty set (the vertex set). A *nested hypergraph* on V is a triple

$$H = (V, E, \rho),$$

where E is a finite set (the set of hyperedges) and

$$\rho : V \uplus E \to \mathbb{N}$$

is a *rank function* such that:

1. $\rho(v) = 0$ for all $v \in V$;
2. for every $e \in E$, the hyperedge e is a nonempty finite set satisfying

$$e \subseteq V \uplus E,$$

   and for every $x \in e$ one has

$$\rho(x) < \rho(e).$$

A hyperedge $e \in E$ is called a *nesting hyperedge* if $e \cap E \neq \emptyset$ (i.e., e contains at least one hyperedge as a member). We say that H is *(nontrivially) nested* if it has at least one nesting hyperedge.

Remark 2.10.2 (Well-foundedness)**.** *The strict inequality $\rho(x) < \rho(e)$ for $x \in e$ excludes membership cycles among hyperedges, so the nesting relation is well-founded.*

Example 2.10.3 (A nested hypergraph with a hyperedge containing another hyperedge)**.** Let the vertex set be

$$V = \{a, b, c\}.$$

Introduce two hyperedges $E = \{e_1, e_2\}$ and define them as elements of $V \uplus E$ by

$$e_1 = \{a, b\} \subseteq V, \qquad e_2 = \{e_1, c\} \subseteq V \uplus E.$$

Define a rank function $\rho : V \uplus E \to \mathbb{N}$ by

$$\rho(a) = \rho(b) = \rho(c) = 0, \qquad \rho(e_1) = 1, \qquad \rho(e_2) = 2.$$

Then for every $x \in e_1 = \{a, b\}$ we have $\rho(x) = 0 < 1 = \rho(e_1)$, and for every $x \in e_2 = \{e_1, c\}$ we have

$$\rho(e_1) = 1 < 2 = \rho(e_2), \qquad \rho(c) = 0 < 2 = \rho(e_2).$$

Hence $H = (V, E, \rho)$ is a nested hypergraph in the sense of Definition 2.10.1. Moreover, e_2 is a nesting hyperedge because $e_2 \cap E = \{e_1\} \neq \emptyset$. An overview diagram of this example is provided in Fig. 2.10.

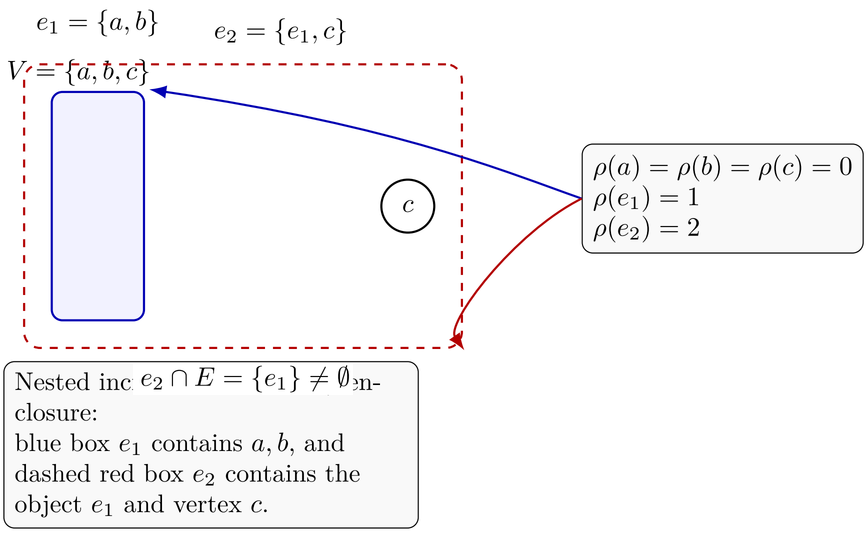


Figure 2.10: A nested hypergraph where the hyperedge e_2 contains another hyperedge e_1.

Definition 2.10.4 (Nested level-n SuperHyperGraph)**.** Fix a finite nonempty base set V_0 and an integer $n \geq 0$. Define iterated powersets recursively by

$$\mathcal{P}^0(V_0) := V_0, \qquad \mathcal{P}^{k+1}(V_0) := \mathcal{P}\big(\mathcal{P}^k(V_0)\big) \quad (k \geq 0).$$

Let

$$V_n \subseteq \mathcal{P}^n(V_0)$$

be a finite set; its elements are called *n-supervertices*. A *nested level-n SuperHyperGraph* is a triple

$$H^{(n)} = (V_n, E_n, \rho),$$

where E_n is a finite set (of superhyperedges) and

$$\rho : V_n \uplus E_n \to \mathbb{N}$$

is a rank function such that:

1. $\rho(v) = 0$ for all $v \in V_n$;
2. for every $e \in E_n$, the superhyperedge e is a nonempty finite set satisfying

   $$e \subseteq V_n \uplus E_n,$$

   and for every $x \in e$ one has

   $$\rho(x) < \rho(e).$$

A superhyperedge $e \in E_n$ is called *simple* if $e \subseteq V_n$, and it is called *nesting* if $e \cap E_n \neq \emptyset$ (i.e., e contains at least one superhyperedge as a member).

Example 2.10.5 (A nested level-1 SuperHyperGraph)**.** Let the base set be

$$V_0 = \{1, 2, 3\},$$

and take $n = 1$. Then $\mathcal{P}^1(V_0) = \mathcal{P}(V_0)$. Define the 1-supervertex set

$$V_1 = \big\{\{1\}, \{2\}, \{1, 2\}, \{3\}\big\} \subseteq \mathcal{P}(V_0).$$

Introduce two superhyperedges $E_1 = \{E_a, E_b\}$ and define them as elements of $V_1 \uplus E_1$ by

$$E_a = \big\{\{1, 2\}, \{3\}\big\} \subseteq V_1, \qquad E_b = \big\{E_a, \{1\}\big\} \subseteq V_1 \uplus E_1.$$

Define a rank function $\rho : V_1 \uplus E_1 \to \mathbb{N}$ by

$$\rho(v) = 0 \ \ (v \in V_1), \qquad \rho(E_a) = 1, \qquad \rho(E_b) = 2.$$

Then $H^{(1)} = (V_1, E_1, \rho)$ is a nested level-1 SuperHyperGraph in the sense of Definition 2.10.4: indeed, each endpoint of E_a has rank $0 < 1$, and each endpoint of E_b has rank < 2. Here E_a is *simple* (since $E_a \subseteq V_1$), while E_b is *nesting* because it contains E_a.

2.11 Multi-Hypergraph and Multi-Superhypergraph

A *Multi-Hypergraph* is a hypergraph that allows repeated hyperedges; edges form a multiset of nonempty vertex subsets with multiplicity counts [114, 115]. It is known that a multi-hypergraph generalizes both hypergraphs and multigraphs. A *Multi-Superhypergraph* is a superhypergraph that allows repeated superedges; vertices are nested-set objects, and superedges carry multiplicities.

Definition 2.11.1 (Multi-Hypergraph)**.** [114, 115] Let V be a finite nonempty set. Write

$$\mathcal{P}^*(V) := \mathcal{P}(V) \setminus \{\emptyset\}.$$

A *multi-hypergraph* is a triple

$$H = (V, E, \mu),$$

where

1. E is a (multi)set of hyperedges, each hyperedge being a nonempty subset of V (i.e. $e \in \mathcal{P}^*(V)$ for all $e \in E$), and

2. $\mu : E \to \mathbb{N}_{>0}$ is a multiplicity function, where $\mu(e)$ is the number of times the hyperedge e occurs.

Equivalently, one may regard E itself as a multiset in which each hyperedge e appears with multiplicity $\mu(e)$.

Example 2.11.2 (Repeated group interactions as a multi-hypergraph)**.** Let

$$V = \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}\}$$

be four participants. Suppose the same three-person meeting $\{\mathsf{A}, \mathsf{B}, \mathsf{C}\}$ occurs three times during a week, while the pair meeting $\{\mathsf{B}, \mathsf{D}\}$ occurs twice. Define the (multi)set of hyperedges

$$E = \{\, e_1 = \{\mathsf{A}, \mathsf{B}, \mathsf{C}\},\ e_2 = \{\mathsf{B}, \mathsf{D}\}\,\} \subseteq \mathcal{P}^*(V)$$

together with the multiplicity function $\mu : E \to \mathbb{N}_{>0}$ given by

$$\mu(e_1) = 3, \qquad \mu(e_2) = 2.$$

Then $H = (V, E, \mu)$ is a multi-hypergraph in the sense of Definition 2.11.1. An overview diagram of this example is provided in Fig. 2.11.

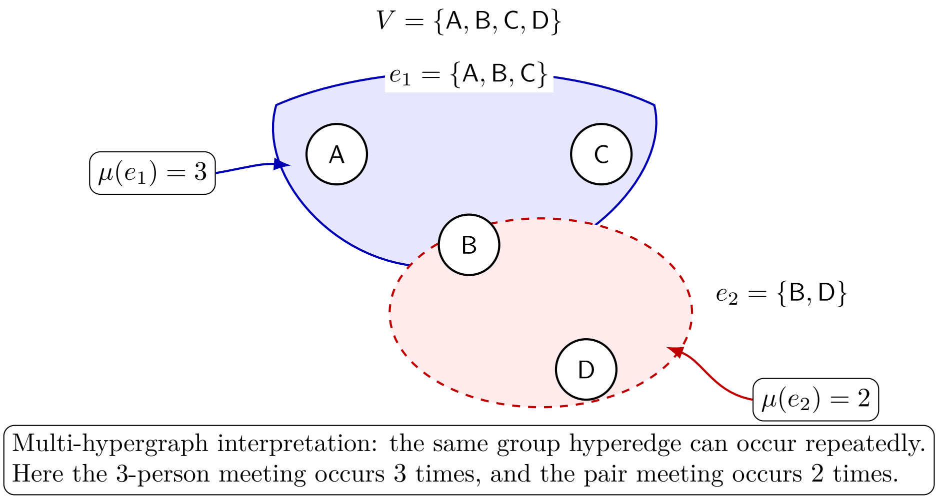


Figure 2.11: A multi-hypergraph for repeated group interactions (Example 2.11.2).

Definition 2.11.3 (Multi n-SuperHyperGraph)**.** Let V_0 be a finite nonempty set and let $n \in \mathbb{N}_0$. Define the iterated powersets by

$$\mathcal{P}^0(V_0) := V_0, \qquad \mathcal{P}^{k+1}(V_0) := \mathcal{P}\big(\mathcal{P}^k(V_0)\big) \quad (k \geq 0).$$

A *Multi n-SuperHyperGraph* is a triple

$$\mathrm{MSHG}^{(n)} = (V, \mathcal{E}, \mu),$$

such that

1. $V \subseteq \mathcal{P}^n(V_0)$ (the set of n-supervertices);

2. $\mathcal{E} \subseteq \mathcal{P}(V) \setminus \{\varnothing\}$ (the set of n-superedges);

3. $\mu : \mathcal{E} \to \mathbb{N}_{>0}$ is a multiplicity function.

For each $e \in \mathcal{E}$, the value $\mu(e)$ indicates that the superedge e occurs $\mu(e)$ times.

Elements of V are called *n-supervertices*, and elements of $\mathcal{E}$ are called *n-superedges*.

Example 2.11.4 (A Multi 1-SuperHyperGraph with repeated superedges)**.** Let the base set be

$$V_0 = \{1, 2, 3, 4\},$$

and take $n = 1$, so $\mathcal{P}^1(V_0) = \mathcal{P}(V_0)$.

Define the 1-supervertex set

$$V = \{\{1,2\},\{3\},\{4\}\} \subseteq \mathcal{P}(V_0).$$

Define two 1-superedges (that is, nonempty subsets of V)

$$e_1 = \{\{1,2\},\{3\}\}, \qquad e_2 = \{\{1,2\},\{4\}\}.$$

Then indeed $e_1, e_2 \in \mathcal{P}(V)$, and hence

$$\mathcal{E} = \{e_1, e_2\} \subseteq \mathcal{P}(V).$$

Assign multiplicities by

$$\mu(e_1) = 2, \qquad \mu(e_2) = 4.$$

Therefore,

$$\mathrm{MSHG}^{(1)} = (V, \mathcal{E}, \mu)$$

is a Multi 1-SuperHyperGraph in the sense of Definition 2.11.3, where e_1 occurs twice and e_2 occurs four times.

2.12 Line Graph and Iterated Line Graph

The *line graph* construction converts edges into vertices: each edge of a graph becomes a vertex in the new graph, and two such vertices are adjacent if and only if the corresponding original edges share a common endpoint [116, 117, 118, 119]. Related notions are also known, including *Fuzzy Line Graphs* [118, 120], Line HyperGraph[121], and *Neutrosophic Line Graphs* [122, 123]. By iterating this operation, one obtains the *iterated line graphs*: starting from $L^0(G) = G$, define $L^{k+1}(G) = L(L^k(G))$, producing a sequence that reflects increasingly higher-order edge-incidence patterns [124, 125, 126, 127, 128, 129]. Recently, this line-graph iteration paradigm has been further extended to *iterated line hypergraphs* and *iterated line superhypergraphs*, and active developments have been reported in these directions [130].

Definition 2.12.1 (Line graph)**.** [117] Let $G = (V, E)$ be a finite simple undirected graph. The *line graph* of G is the graph

$$L(G) := (V_L, E_L)$$

defined by

$$V_L := E, \qquad E_L := \{\{e, f\} \in \tbinom{E}{2} \mid e \cap f \neq \emptyset\}.$$

Equivalently, vertices of $L(G)$ correspond to edges of G, and two vertices are adjacent precisely when the corresponding edges of G share a common endpoint.

Example 2.12.2 (Line graph of a path on four vertices)**.** Let $G = (V, E)$ be the path graph P_4 with

$$V = \{1,2,3,4\}, \qquad E = \{\{1,2\},\{2,3\},\{3,4\}\}.$$

Denote the three edges of G by

$$e_{12} = \{1,2\}, \quad e_{23} = \{2,3\}, \quad e_{34} = \{3,4\}.$$

Then the line graph $L(G) = (V_L, E_L)$ has vertex set

$$V_L = E = \{e_{12}, e_{23}, e_{34}\},$$

and two vertices are adjacent exactly when the corresponding edges in G share an endpoint:

$$E_L = \{\{e_{12}, e_{23}\}, \{e_{23}, e_{34}\}\}.$$

Hence $L(P_4)$ is again a path on three vertices, i.e. $L(P_4) \cong P_3$. This illustrates Definition 2.12.1.

Remark 2.12.3 (Directed variant (line digraph))**.** *If G is directed, one often considers an* oriented *line graph (also called a* line digraph*), whose vertices are the directed edges (arcs) and in which $(u \to v)$ is adjacent to $(v \to w)$, optionally with a non-backtracking constraint $w \neq u$. This oriented construction is closely related to non-backtracking operators used in graph learning.*

Definition 2.12.4 (Iterated line graph)**.** [124, 125] Define the iterated line-graph operator L^k recursively by

$$L^0(G) := G, \qquad L^{k+1}(G) := L\big(L^k(G)\big) \quad (k \in \mathbb{N}).$$

Thus $L^1(G) = L(G)$, $L^2(G) = L(L(G))$, and so forth. The sequence $\{L^k(G)\}_{k \geq 0}$ is called the *line-graph iteration* (or *line-graph hierarchy*) of G.

Table 2.5: Concise overview of a graph, its line graph, and iterated line graphs.

Object	**Vertices**	**Adjacency / meaning**
Graph $G = (V, E)$	Base objects: vertices V.	Edges encode *vertex adjacency*. For a simple undirected graph, $E \subseteq \binom{V}{2}$ and $\{u, v\} \in E$ means u and v are adjacent.
Line graph $L(G)$	Edges of G become vertices: $V(L(G)) := E(G)$.	Two vertices $e, f \in E(G)$ are adjacent in $L(G)$ iff the corresponding edges in G share an endpoint (i.e., $e \cap f \neq \emptyset$). Thus $L(G)$ encodes *edge-incidence adjacency* of G.
Iterated line graph $L^k(G)$	Vertices are edges of the previous iterate: $V(L^k(G)) := E(L^{k-1}(G))$ for $k \geq 1$.	Defined recursively by $L^0(G) = G$ and $L^{k+1}(G) = L(L^k(G))$. This yields a hierarchy capturing progressively higher-order patterns in how edges touch one another across iterations.

Example 2.12.5 (Iterated line graph of a cycle C_4)**.** Let $G = C_4$ be the 4-cycle with vertex set $\{1, 2, 3, 4\}$ and edge set

$$E = \big\{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 1\}\big\}.$$

Each edge in C_4 meets exactly two other edges, so the line graph $L(C_4)$ is again a 4-cycle:

$$L(C_4) \cong C_4.$$

Consequently, the line-graph iteration stabilizes:

$$L^k(C_4) \cong C_4 \qquad \text{for all } k \geq 1.$$

This provides a concrete example of Definition 2.12.4.

For reference, an overview of a graph, its line graph, and iterated line graphs is presented in Table 2.5 (cf.[121]).

2.13 Iterated Total Graph

A total graph has vertices for both vertices and edges of a graph, joining them by adjacency or incidence relationships (cf.[131, 132]).

Definition 2.13.1 (Total graph)**.** [133, 134] Let $G = (V(G), E(G))$ be a loopless simple undirected graph. The *total graph* $T(G)$ is the (simple) graph with

$$V\big(T(G)\big) = V(G) \,\dot\cup\, E(G),$$

where two distinct vertices $x, y \in V(G) \cup E(G)$ are adjacent in $T(G)$ iff one of the following holds:

1. (vertex–vertex) $x, y \in V(G)$ and $xy \in E(G)$;
2. (edge–edge) $x, y \in E(G)$ and $x \cap y \neq \emptyset$ in G;
3. (vertex–edge) $\{x, y\} = \{v, e\}$ with $v \in V(G)$, $e \in E(G)$, and $v \in e$.

Example 2.13.2 (Total graph of the path P_3)**.** Let $G = P_3$ be the path on three vertices

$$V(G) = \{1, 2, 3\}, \qquad E(G) = \{\{1, 2\}, \{2, 3\}\}.$$

Write $e_{12} = \{1, 2\}$ and $e_{23} = \{2, 3\}$. Then the total graph $T(G)$ has vertex set

$$V\big(T(G)\big) = \{1, 2, 3\} \,\dot\cup\, \{e_{12}, e_{23}\}.$$

Adjacency in $T(G)$ arises from the three rules:

- *vertex–vertex:* $1 \sim 2$ and $2 \sim 3$ (since $\{1,2\},\{2,3\} \in E(G)$);
- *edge–edge:* $e_{12} \sim e_{23}$ (since $e_{12} \cap e_{23} = \{2\} \neq \emptyset$ in G);
- *vertex–edge:* $1 \sim e_{12}$, $2 \sim e_{12}$, $2 \sim e_{23}$, $3 \sim e_{23}$ (because each vertex is adjacent to its incident edges).

Thus $T(P_3)$ is a simple graph on five vertices encoding both adjacency and incidence information of P_3, as in the definition of the total graph.

An iterated total graph repeatedly applies the total graph operation to a graph, incorporating vertices, edges, and all incidence relationships at each stage [132, 131].

Definition 2.13.3 (Iterated total graphs)**.** Define $T^0(G) := G$, and for each integer $k \geq 1$ set

$$T^k(G) \;:=\; T\big(T^{k-1}(G)\big).$$

(Thus $T^1(G) = T(G)$, $T^2(G) = T(T(G))$, etc.) This notation is also used in the literature.

Example 2.13.4 (Iterated total graphs starting from K_2)**.** Let $G = K_2$ be the complete graph on two vertices:

$$V(G) = \{1,2\}, \qquad E(G) = \{\{1,2\}\}.$$

Let $e = \{1,2\}$ denote the unique edge. Then the total graph $T(G)$ has vertex set

$$V\big(T(G)\big) = \{1,2,e\}$$

and edges

$$\{1,2\},\ \{1,e\},\ \{2,e\},$$

so $T(K_2) \cong K_3$.

Applying the total operation again yields

$$T^2(K_2) = T\big(T(K_2)\big) = T(K_3).$$

Since K_3 has three vertices and three edges, $T(K_3)$ has 6 vertices and encodes all vertex–vertex, edge–edge, and vertex–edge incidences of K_3. Hence $(T^k(K_2))_{k \geq 0}$ provides a concrete iterated total-graph hierarchy

$$K_2 \;=\; T^0(K_2) \;\longmapsto\; T^1(K_2) \cong K_3 \;\longmapsto\; T^2(K_2) = T(K_3) \;\longmapsto\; \cdots,$$

illustrating the definition of iterated total graphs.

2.14 Hierarchical SuperHyperGraph

A hierarchical superhypergraph is a superhypergraph whose vertices live across multiple powerset levels, with edges allowed to join mixed-level supervertices, while maintaining downward-closure coherence (cf.[135, 136]).

Definition 2.14.1 (Support map and edge-support)**.** Define the support map $\mathrm{supp} : V \to \mathcal{P}(V_0)\backslash\{\emptyset\}$ recursively by

$$\mathrm{supp}(v) := \{v\} \quad (v \in V_0), \qquad \mathrm{supp}(X) := \bigcup_{y \in X} \mathrm{supp}(y) \quad (X \in V \setminus V_0),$$

and for each $e \in E$ define its *base support* (a hyperedge on V_0) by

$$\sigma(e) \;:=\; \bigcup_{x \in e} \mathrm{supp}(x) \;\in\; \mathcal{P}(V_0) \setminus \{\emptyset\}.$$

Definition 2.14.2 (Hierarchical SuperHyperGraph of height r)**.** (cf.[135, 136]) Let V_0 be a finite, nonempty base set. For $k \geq 0$ define iterated powersets

$$\mathcal{P}^0(V_0) := V_0, \qquad \mathcal{P}^{k+1}(V_0) := \mathcal{P}\big(\mathcal{P}^k(V_0)\big),$$

and fix an integer $r \geq 0$. Set the *hierarchical universe*

$$\mathcal{U}_r(V_0) \;:=\; \bigcup_{k=0}^{r} \Big(\mathcal{P}^k(V_0) \setminus \{\emptyset\}\Big).$$

For $x \in \mathcal{U}_r(V_0)$, define its *level*

$$\ell(x) := \min\{ k \in \{0, 1, \dots, r\} : x \in \mathcal{P}^k(V_0) \}.$$

A *hierarchical superhypergraph of height* r on V_0 is a pair

$$\mathbb{H}^{\langle r \rangle} = (V, E)$$

such that

(H1) (*Hierarchical vertex set*) V is a finite nonempty set with

$$V \subseteq \mathcal{U}_r(V_0).$$

Elements of V are called *hierarchical supervertices.*

(H2) (*Cross-level edges*) E is a finite family of nonempty subsets of V:

$$E \subseteq \mathcal{P}(V) \setminus \{\emptyset\}.$$

Elements of E are called *hierarchical superhyperedges.* In particular, a superhyperedge may contain supervertices of *different* levels.

(H3) (*Coherence / downward closure*) If $X \in V$ and $\ell(X) \geq 1$, then

$$X \subseteq V.$$

Equivalently, whenever a higher-level supervertex is present, all its immediate constituents are also present as supervertices.

For each $k \in \{0, \dots, r\}$ we define the *k-th layer* by

$$V_k := \{ x \in V : \ell(x) = k \}, \qquad \text{so that} \qquad V = \dot{\bigcup}_{k=0}^{r} V_k.$$

Example 2.14.3 (A hierarchical SuperHyperGraph of height 2 with cross-level edges)**.** Let the base set be

$$V_0 = \{a, b, c, d\}.$$

Consider the hierarchical universe

$$\mathcal{U}_2(V_0) = \big(V_0\big) \;\cup\; \big(\mathcal{P}(V_0) \setminus \{\emptyset\}\big) \;\cup\; \big(\mathcal{P}(\mathcal{P}(V_0)) \setminus \{\emptyset\}\big).$$

Define hierarchical supervertices (across levels $0, 1, 2$) by

$$X = \{a, b\} \in \mathcal{P}(V_0), \qquad Y = \{c, d\} \in \mathcal{P}(V_0), \qquad U = \{X\} = \{\{a, b\}\} \in \mathcal{P}(\mathcal{P}(V_0)).$$

Set

$$V = \{a, b, c, d,\ X,\ Y,\ U\} \subseteq \mathcal{U}_2(V_0).$$

This V satisfies the coherence condition (H3): since $X, Y \in V$ (level 1), their elements a, b, c, d belong to V; and since $U \in V$ (level 2), its element X belongs to V.

Next, define a family of hierarchical superhyperedges

$$E = \{e_1, e_2\}, \qquad e_1 = \{X, Y\}, \qquad e_2 = \{U, c\}.$$

Thus e_1 links two level-1 supervertices (two teams), while e_2 links a level-2 supervertex (a nested unit) to a level-0 vertex (an individual), illustrating a cross-level edge as allowed by (H2). Therefore

$$\mathbb{H}^{\langle 2 \rangle} = (V, E)$$

is a hierarchical superhypergraph of height 2 in the sense of Definition 2.14.2.

Support map and edge-support. Using Definition 2.14.1, the support map Supp satisfies

$$\begin{aligned}
\operatorname{Supp}(a) = \{a\},\ \operatorname{Supp}(b) = \{b\},\\
\operatorname{Supp}(c) = \{c\},\ \operatorname{Supp}(d) = \{d\},\\
\operatorname{Supp}(X) = \operatorname{Supp}(\{a, b\}) = \{a, b\},\\
\operatorname{Supp}(Y) = \{c, d\},\\
\operatorname{Supp}(U) = \operatorname{Supp}(\{X\}) = \operatorname{Supp}(X) = \{a, b\}.
\end{aligned}$$

Hence the base supports of the two edges are

$$\sigma(e_1) = \operatorname{Supp}(X) \cup \operatorname{Supp}(Y) = \{a, b, c, d\},$$
$$\sigma(e_2) = \operatorname{Supp}(U) \cup \operatorname{Supp}(c) = \{a, b, c\}.$$

An overview diagram of this example is provided in Fig. 2.12.

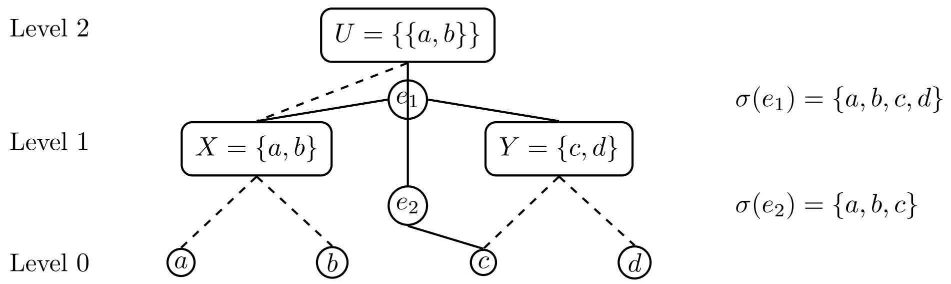


Figure 2.12: A hierarchical superhypergraph of height 2: vertices may live on different levels and edges may cross levels. Dashed lines indicate constituent relations (coherence), while e_1, e_2 are superhyperedges.

2.15 Recursive HyperGraph and Recursive SuperHyperGraph

A *recursive hypergraph* is a hypergraph-like object in which a hyperedge may contain ordinary vertices and also lower-level hyperedges as elements, thereby permitting nested incidence up to a prescribed recursion depth [137, 138, 139, 140].

Definition 2.15.1 (Depth-k powerset universe)**.** [137, 140] Let S be a nonempty set and let $k \in \mathbb{N} \cup \{0\}$. Define a hierarchy of sets $(S_i)_{i \geq 0}$ by

$$S_0 := S, \qquad S_i := \mathcal{P}\Big(\bigcup_{j=0}^{i-1} S_j\Big) \quad (i \geq 1).$$

Set $U_{S,k} := \bigcup_{i=0}^{k} S_i$. The *depth-$k$ powerset universe* over S is

$$2_{S,k} \ := \ \mathcal{P}(U_{S,k}).$$

Definition 2.15.2 (k-recursive hypergraph)**.** [137, 140] Let V be a finite vertex set and let $k \in \mathbb{N} \cup \{0\}$. A *$k$-recursive hypergraph* is a pair

$$H = (V, E)$$

such that

$$E \subseteq 2_{V,k} \setminus \{\emptyset\},$$

where $2_{V,k}$ is the depth-k powerset universe from Definition 2.15.1 applied to $S = V$.

In particular, for $k = 0$ one has $2_{V,0} = \mathcal{P}(V)$ and thus $E \subseteq \mathcal{P}(V) \setminus \{\emptyset\}$, i.e., H reduces to an ordinary hypergraph.

Example 2.15.3 (A 1-recursive hypergraph (edge-of-edges for task bundles))**.** Let

$$V = \{a, b, c, d\}$$

be four tasks. Consider two ordinary hyperedges (task bundles)

$$e_1 = \{a, b\}, \qquad e_2 = \{c, d\}.$$

A 1-*recursive* hyperedge may contain vertices and also lower-level hyperedges as elements. Define

$$e_3 = \{e_1, c\},$$

which represents a higher-level bundle "do the bundle $\{a, b\}$ together with task c." Let

$$E = \{e_1, e_2, e_3\}.$$

Then $H = (V, E)$ is a 1-recursive hypergraph in the sense of Definition 2.15.2: indeed, $e_1, e_2 \in 2_{V,0} = \mathcal{P}(V)$ and $e_3 \in 2_{V,1}$ because it mixes a depth-0 hyperedge e_1 with a vertex c. An overview diagram of this example is provided in Fig. 2.13.

An (n, k)-recursive SuperHyperGraph combines hierarchical supervertices (via iterated powersets) with recursive superhyperedges of bounded depth k, allowing edges to contain supervertices and nested lower-level edges as elements.

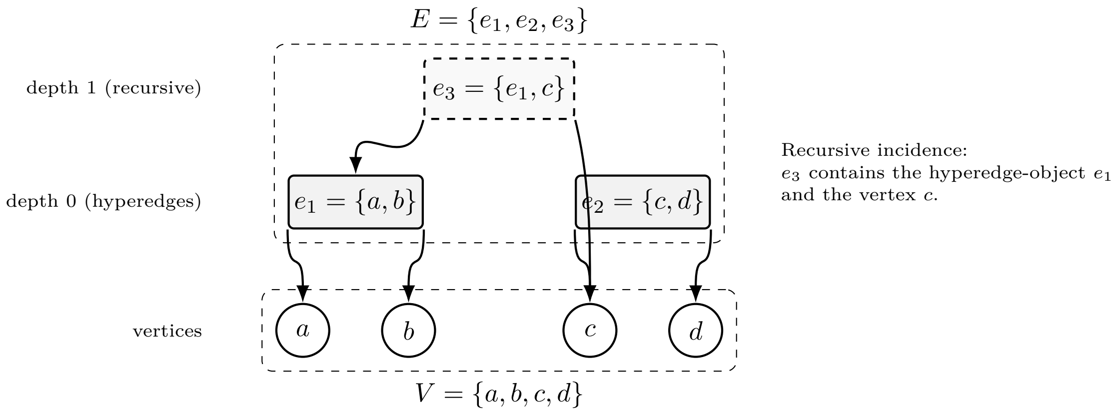


Figure 2.13: A schematic illustration of the 1-recursive hypergraph in Example 2.15.3. The recursive hyperedge e_3 contains a lower-level hyperedge e_1 and a vertex c.

Definition 2.15.4 ((n, k)-recursive SuperHyperGraph)**.** [141, 142, 143] Fix a finite nonempty base set V_0 and let $n, k \in \mathbb{N} \cup \{0\}$. An *$(n, k)$-recursive SuperHyperGraph* is a pair

$$\mathrm{RSHG}^{(n,k)} = (V, E)$$

satisfying:

(i) *(Hierarchical supervertex set).* $V \subseteq \mathcal{P}^n(V_0)$.

(ii) *(Recursive superhyperedge family).* $E \subseteq 2_{V,k} \setminus \{\emptyset\}$, where $2_{V,k}$ is the depth-k powerset universe constructed from $S = V$ as in Definition 2.15.1.

Example 2.15.5 (A $(1, 1)$-recursive SuperHyperGraph (teams with a recursive superedge))**.** Let the base set be

$$V_0 = \{1, 2, 3, 4\},$$

and take $n = 1$, so $\mathcal{P}^1(V_0) = \mathcal{P}(V_0)$. Define the 1-supervertex set (teams)

$$V = \big\{\{1, 2\}, \{3\}, \{4\}\big\} \subseteq \mathcal{P}(V_0).$$

Consider two ordinary superhyperedges (depth 0 edges in $\mathcal{P}(V)$):

$$e_1 = \big\{\{1, 2\}, \{3\}\big\}, \qquad e_2 = \big\{\{1, 2\}, \{4\}\big\}.$$

Now form a recursive superhyperedge of depth 1 by letting

$$e_3 = \{e_1, \{4\}\} \in 2_{V,1},$$

which can be read as a higher-level interaction “activate the collaboration e_1 and also involve the team $\{4\}$.” Let

$$E = \{e_1, e_2, e_3\} \subseteq 2_{V,1} \setminus \{\emptyset\}.$$

Then $\mathrm{RSHG}^{(1,1)} = (V, E)$ is a $(1, 1)$-recursive SuperHyperGraph in the sense of Definition 2.15.4.

2.16 Tree-Vertex Graph

A tree-vertex graph labels tree nodes by nested vertex-sets; edges connect nodes within levels, representing relations across abstraction depths [144].

Definition 2.16.1 (Depth-n Tree-Vertex Graph with level edges)**.** [144] Let V_0 be a finite, nonempty set of *base vertices* and let $n \in \mathbb{N}$. A *depth-n Tree-Vertex Graph (TVG) on V_0* is a quadruple

$$\mathrm{TVG}^{(n)} \;=\; \big(V_0,\; T,\; \eta,\; \{E^{(k)}\}_{k=0}^{n}\big),$$

where:

(i) $T = (\mathcal{N}, \mathcal{F}, r)$ is a rooted tree whose leaves have depth 0 and whose root has depth n. For each $k \in \{0, \dots, n\}$ define the level set

$$\mathcal{N}_k := \{u \in \mathcal{N} \mid \mathrm{depth}(u) = k\}.$$

(ii) η is a *nested labeling* (level-typed label map)

$$\eta : \mathcal{N} \longrightarrow \bigcup_{k=0}^{n} \mathrm{PS}^k(V_0) \setminus \{\emptyset\}$$

satisfying:

(a) (*Level-typing*) For every $u \in \mathcal{N}_k$, one has

$$\eta(u) \in \mathrm{PS}^k(V_0) \setminus \{\emptyset\}.$$

(b) (*Leaf grounding*) The restriction $\eta|_{\mathcal{N}_0} : \mathcal{N}_0 \to V_0$ is a bijection (so each leaf represents exactly one base vertex).

(c) (*Recursive nesting along the tree*) For every internal node $u \in \mathcal{N}_k$ with $k \geq 1$,

$$\eta(u) \;=\; \{\eta(v) \mid v \in \mathrm{Ch}(u)\},$$

so the label of u is literally the set of its children's labels (hence lives in the next iterated powerset).

(iii) The *support* (flattened base-vertex set) of a tree-vertex $u \in \mathcal{N}_k$ is defined by

$$\lambda(u) := \mathrm{Flat}_k(\eta(u)) \subseteq V_0$$

.

(iv) For each level $k \in \{0, \dots, n\}$, $E^{(k)}$ is a set of *level-k graph edges*:

$$E^{(k)} \subseteq \{\{u, v\} \subseteq \mathcal{N}_k \mid u \neq v\}.$$

An edge $\{u, v\} \in E^{(k)}$ encodes a binary relation *within the same abstraction depth* k between the hierarchical units represented by u and v.

Example 2.16.2 (A depth-2 Tree-Vertex Graph for a small organization)**.** Let the base vertices be four employees

$$V_0 = \{a, b, c, d\}.$$

We construct a depth-2 Tree-Vertex Graph $\mathrm{TVG}^{(2)}$ in the sense of Definition 2.16.1.

(1) The rooted tree. Let $T = (\mathcal{N}, \mathcal{F}, r)$ be the rooted tree with node set

$$\mathcal{N} = \{r, u_1, u_2, a_0, b_0, c_0, d_0\},$$

root r at depth 2, internal nodes u_1, u_2 at depth 1, and leaves

$$\mathcal{N}_0 = \{a_0, b_0, c_0, d_0\}, \quad \mathcal{N}_1 = \{u_1, u_2\}, \quad \mathcal{N}_2 = \{r\}.$$

Define the parent–child relations by

$$\mathrm{Ch}(r) = \{u_1, u_2\}, \qquad \mathrm{Ch}(u_1) = \{a_0, b_0\}, \qquad \mathrm{Ch}(u_2) = \{c_0, d_0\}.$$

Thus the tree groups employees into two teams $\{a, b\}$ and $\{c, d\}$, and then into one department.

(2) Nested labeling. Define $\eta : \mathcal{N} \to \bigcup_{k=0}^{2} \mathrm{PS}^k(V_0) \setminus \{\emptyset\}$ by

$$\eta(a_0) = a,\ \eta(b_0) = b,\ \eta(c_0) = c,\ \eta(d_0) = d,$$

$$\eta(u_1) = \{\eta(a_0), \eta(b_0)\} = \{a, b\}, \qquad \eta(u_2) = \{\eta(c_0), \eta(d_0)\} = \{c, d\},$$

$$\eta(r) = \{\eta(u_1), \eta(u_2)\} = \{\{a, b\}, \{c, d\}\}.$$

By construction, $\eta|_{\mathcal{N}_0}$ is a bijection onto V_0, and each internal label is exactly the set of its children's labels.

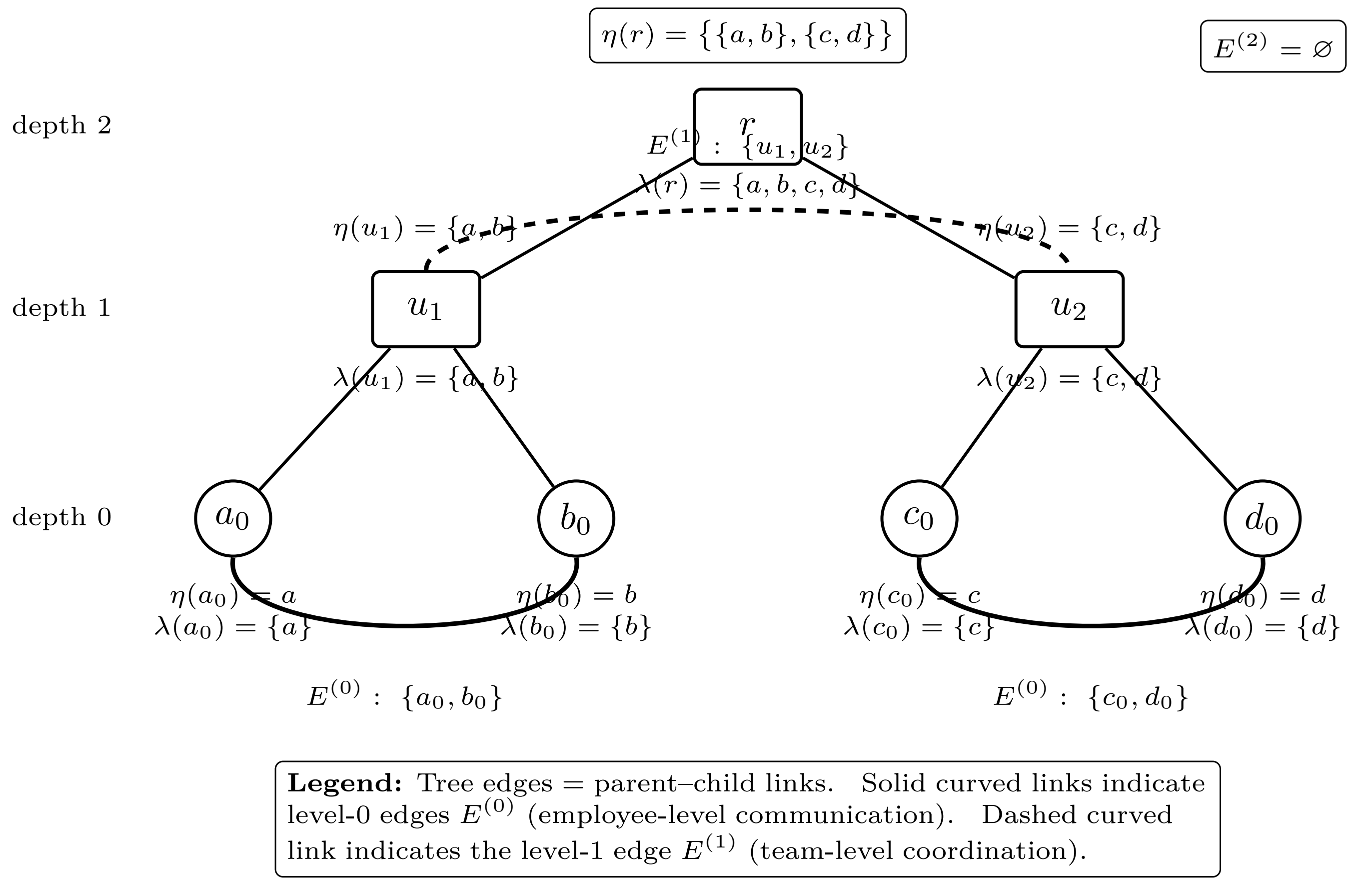


Figure 2.14: A depth-2 Tree-Vertex Graph for a small organization (Example 2.16.2). Leaves represent employees, internal nodes represent teams, and the root represents the department.

(3) Supports. Let $\lambda(u) = \mathrm{Flat}_k(\eta(u)) \subseteq V_0$ be the flattened support. Then

$$\lambda(a_0) = \{a\},\ \lambda(b_0) = \{b\},\ \lambda(c_0) = \{c\},\ \lambda(d_0) = \{d\},$$

$$\lambda(u_1) = \{a, b\}, \qquad \lambda(u_2) = \{c, d\}, \qquad \lambda(r) = \{a, b, c, d\}.$$

(4) Level edges. Define level-k edge sets by

$$E^{(0)} = \big\{\{a_0, b_0\}, \{c_0, d_0\}\big\}, \qquad E^{(1)} = \big\{\{u_1, u_2\}\big\}, \qquad E^{(2)} = \emptyset.$$

Here $E^{(0)}$ encodes pairwise communication links inside each team at the employee level, while $E^{(1)}$ encodes a coordination link between the two teams.

Therefore,

$$\mathrm{TVG}^{(2)} = \big(V_0,\ T,\ \eta,\ \{E^{(k)}\}_{k=0}^{2}\big)$$

is a depth-2 Tree-Vertex Graph on V_0. An overview diagram of this example is provided in Fig. 2.14.

2.17 Tensor network graph

A tensor network graph assigns tensors to vertices and index bonds to edges; contracting edges sums shared indices, yielding a tensor [145, 146]. A tensor network graph encodes higher-order interactions as tensors on nodes, with edges denoting index contractions, representing multiway correlations compactly for efficient computation and inference.

Definition 2.17.1 (Tensor network graph)**.** A *tensor network graph* is a triple

$$\mathcal{N} = (G,\ (T_v)_{v \in V(G)},\ (\mathcal{I}_e)_{e \in E(G)}),$$

where G is a finite (multi)graph, each vertex $v \in V(G)$ is assigned a tensor

$$T_v \in \bigotimes_{e \in \mathrm{Inc}(v)} \mathbb{K}^{\mathcal{I}_e}$$

over a field $\mathbb{K}$ (typically $\mathbb{R}$ or $\mathbb{C}$), and each edge $e \in E(G)$ is assigned a finite index set $\mathcal{I}_e$ (equivalently, a bond dimension $|\mathcal{I}_e|$). Here $\mathrm{Inc}(v)$ denotes the multiset of edges incident to v; half-edges (dangling edges) represent *open indices* (external legs). The *value* (or *contraction*) of $\mathcal{N}$ is the tensor obtained by summing over all internal indices corresponding to non-dangling edges, i.e. contracting the paired index spaces $\mathbb{K}^{\mathcal{I}_e} \otimes \mathbb{K}^{\mathcal{I}_e}$ along each internal edge e.

Example 2.17.2 (A tensor network graph for a length-3 matrix product state)**.** Let $\mathbb{K} = \mathbb{R}$. Consider a path graph on three vertices

$$G: \quad v_1 \; - \; v_2 \; - \; v_3,$$

with two internal edges e_{12} and e_{23}, and with one dangling half-edge attached to each vertex (representing a physical/open index). Assign index sets

$$\mathcal{I}_{12} = \{1, \dots, D\}, \qquad \mathcal{I}_{23} = \{1, \dots, D\}$$

to the internal edges (bond dimension D), and assign

$$\mathcal{I}_1 = \{1, \dots, d\}, \qquad \mathcal{I}_2 = \{1, \dots, d\}, \qquad \mathcal{I}_3 = \{1, \dots, d\}$$

to the three dangling edges (physical dimension d).

Assign tensors

$$T_{v_1} \in \mathbb{R}^{d \times D}, \qquad T_{v_2} \in \mathbb{R}^{D \times d \times D}, \qquad T_{v_3} \in \mathbb{R}^{D \times d}$$

so that T_{v_1} has indices (i_1, α), T_{v_2} has indices (α, i_2, β), and T_{v_3} has indices (β, i_3) with $i_j \in \{1, \dots, d\}$ and $\alpha, \beta \in \{1, \dots, D\}$. Then the tensor network contraction produces a 3-way tensor

$$\Psi \in \mathbb{R}^{d \times d \times d}$$

whose entries are given by

$$\Psi_{i_1 i_2 i_3} = \sum_{\alpha=1}^{D} \sum_{\beta=1}^{D} \left(T_{v_1}\right)_{i_1, \alpha} \left(T_{v_2}\right)_{\alpha, i_2, \beta} \left(T_{v_3}\right)_{\beta, i_3}.$$

Thus $\mathcal{N} = (G, (T_v)_{v \in V(G)}, (\mathcal{I}_e)_{e \in E(G)})$ is a tensor network graph in the sense of Definition 2.17.1; the internal edges e_{12}, e_{23} are contracted (summed over), while the three dangling edges remain as open indices.

2.18 MultiTensor and Iterated MultiTensor Network

MultiTensor Network assigns each node a finite multiset of compatible local tensors, generating multiple weighted contractions through realization choices while preserving one external tensor space. Iterated MultiTensor Network assigns nodes iterated finite multisets of compatible local tensors, recursively flattening multiplicities into effective multitensor assignments before canonically computing weighted network contractions.

Definition 2.18.1 (Finite multiset)**.** Let X be a set. A *finite multiset* on X is a function

$$m : X \to \mathbb{N}_0$$

with finite support

$$\mathrm{supp}(m) := \{x \in X : m(x) > 0\}.$$

We write

$$\mathcal{M}_{\mathrm{fin}}(X)$$

for the set of all finite multisets on X. If $x \in X$, then $m(x)$ is called the *multiplicity* of x in m.

Definition 2.18.2 (MultiTensor network graph)**.** Let

$$\mathcal{T}_v := \bigotimes_{e \in \mathrm{Inc}(v)} \mathbb{K}^{\mathcal{I}_e}$$

denote the local tensor space associated with a vertex v of a finite graph G, where $\mathbb{K}$ is a field and each edge or half-edge e is assigned a finite index set $\mathcal{I}_e$.

A *MultiTensor network graph* is a triple

$$\mathcal{N}^{\mathrm{multi}} = \big(G,\ (M_v)_{v \in V(G)},\ (\mathcal{I}_e)_{e \in E(G)}\big),$$

such that:

- G is a finite (multi)graph, possibly with dangling half-edges;
- for each edge e, $\mathcal{I}_e$ is a finite index set;
- for each vertex $v \in V(G)$,
$$M_v \in \mathcal{M}_{\mathrm{fin}}(\mathcal{T}_v)$$
is a finite multiset of tensors of the same local type $\mathcal{T}_v$.

A *realization* of $\mathcal{N}^{\mathrm{multi}}$ is a choice

$$\tau = (T_v)_{v \in V(G)}$$

such that

$$T_v \in \mathrm{supp}(M_v) \qquad (\forall\, v \in V(G)).$$

Its *multiplicity* is defined by

$$\mu(\tau) := \prod_{v \in V(G)} M_v(T_v).$$

For each realization τ, its *contraction*

$$\mathrm{Contr}(\tau)$$

is the tensor obtained by contracting the ordinary tensor network graph

$$\big(G,\ (T_v)_{v \in V(G)},\ (\mathcal{I}_e)_{e \in E(G)}\big)$$

in the sense of Definition 2.17.1.

The *contraction multiset* of $\mathcal{N}^{\mathrm{multi}}$ is the finite multiset of output tensors defined by

$$\mathrm{Contr}_{\mathrm{multi}}\big(\mathcal{N}^{\mathrm{multi}}\big)(X) := \sum_{\substack{\tau \text{ realization} \\ \mathrm{Contr}(\tau) = X}} \mu(\tau),$$

for each output tensor X in the external tensor space determined by the dangling half-edges.

If desired, one may also define the *aggregated contraction* by

$$\mathrm{AContr}\big(\mathcal{N}^{\mathrm{multi}}\big) := \sum_{\tau \text{ realization}} \mu(\tau)\ \mathrm{Contr}(\tau),$$

provided all output tensors belong to the same external tensor space.

Definition 2.18.3 (Iterated finite multiset)**.** For a set X, define recursively

$$\mathcal{M}^0_{\mathrm{fin}}(X) := X, \qquad \mathcal{M}^{n+1}_{\mathrm{fin}}(X) := \mathcal{M}_{\mathrm{fin}}\big(\mathcal{M}^n_{\mathrm{fin}}(X)\big) \qquad (n \geq 0).$$

Thus, an element of $\mathcal{M}^n_{\mathrm{fin}}(X)$ is an *n-fold iterated finite multiset* over X.

Example 2.18.4 (A concrete MultiTensor network graph)**.** Let $\mathbb{K} = \mathbb{R}$, and let G be the graph with two vertices

$$V(G) = \{v_1, v_2\}$$

and one internal edge

$$E(G) = \{e\}, \qquad e = \{v_1, v_2\}.$$

Assign the index set

$$\mathcal{I}_e = \{1,2\}.$$

Since the only incident edge at each vertex is e, the local tensor spaces are

$$\mathcal{T}_{v_1} = \mathbb{R}^{\mathcal{I}_e}, \qquad \mathcal{T}_{v_2} = \mathbb{R}^{\mathcal{I}_e}.$$

Thus each local tensor may be identified with a vector in $\mathbb{R}^2$.

Define

$$A_1 = (1,0), \qquad A_2 = (0,1) \in \mathbb{R}^2,$$

and

$$B_1 = (1,1), \qquad B_2 = (1,-1) \in \mathbb{R}^2.$$

Now assign finite multisets of local tensors by

$$M_{v_1} = 2[A_1] + 1[A_2], \qquad M_{v_2} = 1[B_1] + 3[B_2].$$

Equivalently,

$$M_{v_1}(A_1) = 2, \quad M_{v_1}(A_2) = 1, \qquad M_{v_2}(B_1) = 1, \quad M_{v_2}(B_2) = 3,$$

and all other multiplicities are zero.

Hence

$$\mathcal{N}^{\text{multi}} = \big(G, (M_v)_{v\in V(G)}, (\mathcal{I}_e)_{e\in E(G)}\big)$$

is a MultiTensor network graph.

A realization of $\mathcal{N}^{\text{multi}}$ is a choice

$$\tau = (T_{v_1}, T_{v_2})$$

with

$$T_{v_1} \in \{A_1, A_2\}, \qquad T_{v_2} \in \{B_1, B_2\}.$$

Thus there are exactly four realizations:

$$\tau_1 = (A_1, B_1), \qquad \tau_2 = (A_1, B_2), \qquad \tau_3 = (A_2, B_1), \qquad \tau_4 = (A_2, B_2).$$

Since the network has no dangling half-edges, each contraction is a scalar. More precisely,

$$\mathrm{Contr}(\tau) = \sum_{i\in\mathcal{I}_e} T_{v_1}(i)\, T_{v_2}(i),$$

which is just the standard dot product in $\mathbb{R}^2$.

We compute:

$$\mathrm{Contr}(\tau_1) = A_1 \cdot B_1 = (1,0)\cdot(1,1) = 1,$$
$$\mathrm{Contr}(\tau_2) = A_1 \cdot B_2 = (1,0)\cdot(1,-1) = 1,$$
$$\mathrm{Contr}(\tau_3) = A_2 \cdot B_1 = (0,1)\cdot(1,1) = 1,$$
$$\mathrm{Contr}(\tau_4) = A_2 \cdot B_2 = (0,1)\cdot(1,-1) = -1.$$

Their multiplicities are

$$\mu(\tau_1) = M_{v_1}(A_1)M_{v_2}(B_1) = 2\cdot 1 = 2,$$
$$\mu(\tau_2) = M_{v_1}(A_1)M_{v_2}(B_2) = 2\cdot 3 = 6,$$
$$\mu(\tau_3) = M_{v_1}(A_2)M_{v_2}(B_1) = 1\cdot 1 = 1,$$
$$\mu(\tau_4) = M_{v_1}(A_2)M_{v_2}(B_2) = 1\cdot 3 = 3.$$

Therefore the contraction multiset is

$$\mathrm{Contr}_{\text{multi}}(\mathcal{N}^{\text{multi}}) = 9[1] + 3[-1],$$

because the scalar value 1 occurs with total multiplicity

$$2 + 6 + 1 = 9,$$

whereas the scalar value -1 occurs with multiplicity

$$3.$$

The aggregated contraction is

$$\mathrm{AContr}\big(\mathcal{N}^{\mathrm{multi}}\big) = \sum_{k=1}^{4} \mu(\tau_k)\ \mathrm{Contr}(\tau_k) = 2\cdot 1 + 6\cdot 1 + 1\cdot 1 + 3\cdot(-1) = 6.$$

Thus this example explicitly illustrates how a MultiTensor network graph produces a finite multiset of contraction values, together with an aggregated contraction.

Moreover, each local multiset

$$M_{v_i} \in \mathcal{M}_{\mathrm{fin}}(\mathcal{T}_{v_i}) = \mathcal{M}^1_{\mathrm{fin}}(\mathcal{T}_{v_i}),$$

so this example is also the first-level case of the iterated finite multiset formalism.

Definition 2.18.5 (Flattening of iterated multisets)**.** Let X be a set. For each $n \geq 0$, define recursively a map

$$\mathrm{Flat}_n : \mathcal{M}^n_{\mathrm{fin}}(X) \to \mathcal{M}_{\mathrm{fin}}(X)$$

as follows:

- for $n = 0$, set
$$\mathrm{Flat}_0(x) := \delta_x \qquad (x \in X),$$
where δ_x denotes the singleton multiset supported at x with multiplicity 1;
- for $n \geq 0$ and $M \in \mathcal{M}^{n+1}_{\mathrm{fin}}(X)$, define
$$\mathrm{Flat}_{n+1}(M)(x) := \sum_{A\in \mathrm{supp}(M)} M(A)\ \mathrm{Flat}_n(A)(x) \qquad (x \in X).$$

Hence Flat_n converts an iterated multiset into an ordinary finite multiset on X by recursively collecting multiplicities.

Definition 2.18.6 (n-Iterated MultiTensor network graph)**.** Let $n \geq 0$. For each vertex $v \in V(G)$, let

$$\mathcal{T}_v := \bigotimes_{e\in \mathrm{Inc}(v)} \mathbb{K}^{\mathcal{I}_e}$$

be the corresponding local tensor space.

An *n-Iterated MultiTensor network graph* is a triple

$$\mathcal{N}^{(n)} = \big(G,\ (\Theta_v)_{v\in V(G)},\ (\mathcal{I}_e)_{e\in E(G)}\big),$$

such that:

- G is a finite (multi)graph, possibly with dangling half-edges;
- for each edge e, $\mathcal{I}_e$ is a finite index set;
- for each vertex $v \in V(G)$,
$$\Theta_v \in \mathcal{M}^n_{\mathrm{fin}}(\mathcal{T}_v).$$

Its *effective local multiset* at v is defined by

$$M^{\mathrm{eff}}_v := \mathrm{Flat}_n(\Theta_v) \in \mathcal{M}_{\mathrm{fin}}(\mathcal{T}_v).$$

The *effective MultiTensor network graph* associated with $\mathcal{N}^{(n)}$ is

$$\mathcal{N}^{\mathrm{eff}} := \big(G,\ (M^{\mathrm{eff}}_v)_{v\in V(G)},\ (\mathcal{I}_e)_{e\in E(G)}\big).$$

The *contraction multiset* of $\mathcal{N}^{(n)}$ is then defined by

$$\mathrm{Contr}_{\mathrm{iter}}\big(\mathcal{N}^{(n)}\big) := \mathrm{Contr}_{\mathrm{multi}}\big(\mathcal{N}^{\mathrm{eff}}\big),$$

and, whenever meaningful, its *aggregated contraction* is defined by

$$\mathrm{AContr}\big(\mathcal{N}^{(n)}\big) := \mathrm{AContr}\big(\mathcal{N}^{\mathrm{eff}}\big).$$

In particular:

$$n = 0 \implies \mathcal{N}^{(0)} \text{ is an ordinary tensor network graph,}$$

and

$$n = 1 \implies \mathcal{N}^{(1)} \text{ is exactly a MultiTensor network graph.}$$

Theorem 2.18.7 (Well-definedness of MultiTensor network graph)**.** *Let*

$$\mathcal{N}^{\mathrm{multi}} = \big(G,\ (M_v)_{v\in V(G)},\ (\mathcal{I}_e)_{e\in E(G)}\big)$$

be a MultiTensor network graph in the sense of Definition 2.18.2. Then:

1. *the set of realizations of $\mathcal{N}^{\mathrm{multi}}$ is finite;*
2. *for every realization*
$$\tau = (T_v)_{v\in V(G)} \quad \textit{with} \quad T_v \in \operatorname{supp}(M_v),$$
the ordinary tensor network contraction
$$\operatorname{Contr}(\tau)$$
is well-defined;
3. *the contraction multiset*
$$\operatorname{Contr}_{\mathrm{multi}}\big(\mathcal{N}^{\mathrm{multi}}\big)$$
is well-defined;
4. *the aggregated contraction*
$$\operatorname{AContr}\big(\mathcal{N}^{\mathrm{multi}}\big) = \sum_{\tau} \mu(\tau)\ \operatorname{Contr}(\tau)$$
is well-defined.

Proof. For each vertex $v \in V(G)$, the multiset M_v has finite support by definition, and $V(G)$ is finite. Hence the set of realizations

$$\prod_{v\in V(G)} \operatorname{supp}(M_v)$$

is finite.

Moreover, every tensor chosen at v lies in the same local tensor space

$$\mathcal{T}_v = \bigotimes_{e\in \operatorname{Inc}(v)} \mathbb{K}^{\mathcal{I}_e},$$

so any realization τ determines an ordinary tensor network graph with the same underlying graph G and the same index sets $(\mathcal{I}_e)_{e\in E(G)}$. Therefore its contraction $\operatorname{Contr}(\tau)$ is well-defined in the sense of Definition 2.17.1.

Since the external tensor space is determined only by the dangling half-edges of G and the index sets $\mathcal{I}_e$, all tensors $\operatorname{Contr}(\tau)$ belong to one and the same output space. Because there are only finitely many realizations and each multiplicity

$$\mu(\tau) = \prod_{v\in V(G)} M_v(T_v)$$

is a well-defined nonnegative integer, both

$$\operatorname{Contr}_{\mathrm{multi}}\big(\mathcal{N}^{\mathrm{multi}}\big)$$

and

$$\operatorname{AContr}\big(\mathcal{N}^{\mathrm{multi}}\big)$$

are well-defined. □

Theorem 2.18.8 (Well-definedness of Iterated MultiTensor network graph)**.** *Let*

$$\mathcal{N}^{(n)} = \big(G,\ (\Theta_v)_{v\in V(G)},\ (\mathcal{I}_e)_{e\in E(G)}\big)$$

be an n-Iterated MultiTensor network graph in the sense of Definition 2.18.6. Then:

1. *for each vertex $v \in V(G)$), the flattening*
$$\operatorname{Flat}_n(\Theta_v) \in \mathcal{M}_{\mathrm{fin}}(\mathcal{T}_v)$$
is well-defined;

2. *the effective network*
$$\mathcal{N}^{\text{eff}} = \big(G,\ (\text{Flat}_n(\Theta_v))_{v\in V(G)},\ (\mathcal{I}_e)_{e\in E(G)}\big)$$
is a well-defined MultiTensor network graph;

3. *consequently, both*
$$\text{Contr}_{\text{iter}}\big(\mathcal{N}^{(n)}\big) \quad \textit{and} \quad \text{AContr}\big(\mathcal{N}^{(n)}\big)$$
are well-defined.

Proof. We first show that Flat_n is well-defined for every $n \geq 0$ by induction on n. For $n = 0$, the map
$$\text{Flat}_0(x) = \delta_x$$
is well-defined. Assume that Flat_n is well-defined. Let
$$M \in \mathcal{M}^{n+1}_{\text{fin}}(X).$$
Since M has finite support and each $\text{Flat}_n(A)$ is a finite multiset on X, the formula
$$\text{Flat}_{n+1}(M)(x) = \sum_{A\in \text{supp}(M)} M(A)\ \text{Flat}_n(A)(x)$$
defines a finite-support function $X \to \mathbb{N}_0$. Hence $\text{Flat}_{n+1}(M) \in \mathcal{M}_{\text{fin}}(X)$ is well-defined.

Applying this with $X = \mathcal{T}_v$ for each vertex v, we obtain
$$\text{Flat}_n(\Theta_v) \in \mathcal{M}_{\text{fin}}(\mathcal{T}_v),$$
so the effective network $\mathcal{N}^{\text{eff}}$ is a MultiTensor network graph. Therefore Theorem 2.18.7 applies, and the contraction multiset and aggregated contraction of $\mathcal{N}^{\text{eff}}$ are well-defined. By definition,
$$\text{Contr}_{\text{iter}}\big(\mathcal{N}^{(n)}\big) = \text{Contr}_{\text{multi}}\big(\mathcal{N}^{\text{eff}}\big), \qquad \text{AContr}\big(\mathcal{N}^{(n)}\big) = \text{AContr}\big(\mathcal{N}^{\text{eff}}\big),$$
hence both are well-defined. □

2.19 Tensor Hypernetwork and Tensor Superhypernetwork

A Tensor Hypernetwork is a hypergraph-based tensor network where each vertex carries a tensor and each hyperedge links multiple tensors through shared index contractions simultaneously. A Tensor Superhypernetwork generalizes a tensor hypernetwork by replacing vertices with hierarchical supervertices, enabling tensor contractions over nested, multi-level, higher-order relational structures and systems formally.

Definition 2.19.1 (Tensor Hypernetwork)**.** Let
$$H = (V, E)$$
be a finite hypergraph, where
$$E \subseteq \mathcal{P}(V) \setminus \{\emptyset\}.$$
For each hyperedge $e \in E$, fix a positive integer d_e, and write
$$[d_e] := \{1, 2, \ldots, d_e\}.$$
For each vertex $v \in V$, define its incident-edge set by
$$I(v) := \{\, e \in E \mid v \in e \,\}.$$
A *tensor hypernetwork* over a field $\mathbb{K}$ on H is a tuple
$$\mathcal{N} = \big(H, (d_e)_{e\in E}, (T_v)_{v\in V}\big),$$
where each local tensor is a function
$$T_v : \prod_{e\in I(v)} [d_e] \to \mathbb{K}.$$
If $I(v) = \emptyset$, the empty product is interpreted as a singleton, so $T_v \in \mathbb{K}$.

The associated closed contraction (network value) is
$$Z(\mathcal{N}) := \sum_{(a_e)_{e\in E}\in \prod_{e\in E}[d_e]} \prod_{v\in V} T_v\big((a_e)_{e\in I(v)}\big) \in \mathbb{K}.$$

Definition 2.19.2 (Tensor n-SuperHyperNetwork)**.** Let V_0 be a finite nonempty base set, and define iterated powersets by

$$\mathcal{P}^0(V_0) := V_0, \qquad \mathcal{P}^{k+1}(V_0) := \mathcal{P}\big(\mathcal{P}^k(V_0)\big) \quad (k \geq 0).$$

Let

$$SHG^{(n)} = (V, E)$$

be an n-SuperHyperGraph on V_0, that is,

$$V \subseteq \mathcal{P}^n(V_0), \qquad E \subseteq \mathcal{P}(V) \setminus \{\emptyset\}.$$

Elements of V are called n-supervertices, and elements of E are called n-superedges.

For each superedge $e \in E$, fix a positive integer d_e, and write

$$[d_e] := \{1, 2, \ldots, d_e\}.$$

For each supervertex $x \in V$, define

$$I(x) := \{\, e \in E \mid x \in e \,\}.$$

A *tensor n-SuperHyperNetwork* over a field $\mathbb{K}$ is a tuple

$$\mathcal{N}^{(n)} = \big(V_0; V, E, (d_e)_{e \in E}, (T_x)_{x \in V}\big),$$

where each local supertensor is a function

$$T_x : \prod_{e \in I(x)} [d_e] \to \mathbb{K} \qquad (x \in V).$$

If $I(x) = \emptyset$, the empty product is interpreted as a singleton, so $T_x \in \mathbb{K}$.

Its associated closed contraction is

$$Z(\mathcal{N}^{(n)}) := \sum_{(a_e)_{e \in E} \in \prod_{e \in E} [d_e]} \prod_{x \in V} T_x\big((a_e)_{e \in I(x)}\big) \in \mathbb{K}.$$

A *tensor SuperHyperNetwork* is a tensor n-SuperHyperNetwork for some $n \geq 1$.

Example 2.19.3 (A concrete tensor 1-SuperHyperNetwork)**.** Let the base set be

$$V_0 = \{a, b, c\}.$$

Since

$$\mathcal{P}^1(V_0) = \mathcal{P}(V_0),$$

we may choose 1-supervertices as subsets of V_0. Define

$$x_1 := \{a, b\}, \qquad x_2 := \{b, c\}, \qquad x_3 := \{a, c\},$$

and set

$$V = \{x_1, x_2, x_3\} \subseteq \mathcal{P}(V_0).$$

Now define two 1-superedges by

$$e_1 := \{x_1, x_2\}, \qquad e_2 := \{x_2, x_3\}.$$

Thus

$$E = \{e_1, e_2\} \subseteq \mathcal{P}(V) \setminus \{\emptyset\}.$$

Hence

$$SHG^{(1)} = (V, E)$$

is a 1-SuperHyperGraph on V_0.

Assign bond dimensions

$$d_{e_1} = 2, \qquad d_{e_2} = 2,$$

so that

$$[d_{e_1}] = [d_{e_2}] = \{1, 2\}.$$

For each supervertex, the incident superedge set is

$$I(x_1) = \{e_1\}, \qquad I(x_2) = \{e_1, e_2\}, \qquad I(x_3) = \{e_2\}.$$

Let $\mathbb{K} = \mathbb{R}$. Define the local supertensors as follows:

$$T_{x_1} : [2] \to \mathbb{R}, \qquad T_{x_1}(1) = 1, \quad T_{x_1}(2) = 2,$$

$$T_{x_2} : [2] \times [2] \to \mathbb{R}, \qquad \big(T_{x_2}(i,j)\big)_{i,j=1}^{2} = \begin{pmatrix} 1 & 0 \\ 3 & 1 \end{pmatrix},$$

that is,

$$T_{x_2}(1,1) = 1, \quad T_{x_2}(1,2) = 0, \quad T_{x_2}(2,1) = 3, \quad T_{x_2}(2,2) = 1,$$

and

$$T_{x_3} : [2] \to \mathbb{R}, \qquad T_{x_3}(1) = 2, \quad T_{x_3}(2) = 1.$$

Therefore,

$$\mathcal{N}^{(1)} = \big(V_0; V, E, (d_e)_{e \in E}, (T_x)_{x \in V}\big)$$

is a tensor 1-SuperHyperNetwork over $\mathbb{R}$.

Its closed contraction is

$$Z(\mathcal{N}^{(1)}) = \sum_{(a_{e_1}, a_{e_2}) \in [2] \times [2]} T_{x_1}(a_{e_1})\, T_{x_2}(a_{e_1}, a_{e_2})\, T_{x_3}(a_{e_2}).$$

Writing $i := a_{e_1}$ and $j := a_{e_2}$, we obtain

$$Z(\mathcal{N}^{(1)}) = \sum_{i=1}^{2} \sum_{j=1}^{2} T_{x_1}(i)\, T_{x_2}(i,j)\, T_{x_3}(j).$$

Substituting the above values gives

$$\begin{aligned} Z(\mathcal{N}^{(1)}) &= T_{x_1}(1)T_{x_2}(1,1)T_{x_3}(1) + T_{x_1}(1)T_{x_2}(1,2)T_{x_3}(2) \\ &\quad + T_{x_1}(2)T_{x_2}(2,1)T_{x_3}(1) + T_{x_1}(2)T_{x_2}(2,2)T_{x_3}(2) \\ &= 1 \cdot 1 \cdot 2 + 1 \cdot 0 \cdot 1 + 2 \cdot 3 \cdot 2 + 2 \cdot 1 \cdot 1 \\ &= 2 + 0 + 12 + 2 \\ &= 16. \end{aligned}$$

Hence this example gives an explicit tensor 1-SuperHyperNetwork whose closed contraction equals

$$Z(\mathcal{N}^{(1)}) = 16.$$

2.20 Tensor Train

A tensor train decomposition represents a high-order tensor as a chain product of third-order core tensors connected by auxiliary bond indices [147, 148, 149].

Definition 2.20.1 (Tensor Train (TT) decomposition)**.** [147, 148, 149] Let $d \geq 2$, let

$$A \in \mathbb{F}^{n_1 \times n_2 \times \cdots \times n_d}, \qquad \mathbb{F} \in \{\mathbb{R}, \mathbb{C}\},$$

be a d-th order tensor, and let

$$r = (r_0, r_1, \ldots, r_d) \in \mathbb{N}^{d+1}$$

satisfy

$$r_0 = r_d = 1.$$

We say that A admits a *tensor train decomposition* of TT-rank r if there exist third-order tensors

$$G^{(k)} \in \mathbb{F}^{r_{k-1} \times n_k \times r_k}, \qquad k = 1, 2, \ldots, d,$$

called the *TT-cores*, such that for every multi-index

$$(i_1, i_2, \ldots, i_d) \in \{1, \ldots, n_1\} \times \cdots \times \{1, \ldots, n_d\},$$

the entry of A is represented as

$$A_{i_1 i_2 \cdots i_d} = \sum_{\alpha_1=1}^{r_1} \cdots \sum_{\alpha_{d-1}=1}^{r_{d-1}} G^{(1)}_{1\, i_1\, \alpha_1} G^{(2)}_{\alpha_1\, i_2\, \alpha_2} \cdots G^{(d)}_{\alpha_{d-1}\, i_d\, 1}.$$

Equivalently, for each k and each $i_k \in \{1, \dots, n_k\}$, define the slice

$$G^{(k)}(i_k) \in \mathbb{F}^{r_{k-1} \times r_k}, \qquad \left(G^{(k)}(i_k)\right)_{\alpha_{k-1}, \alpha_k} := G^{(k)}_{\alpha_{k-1}\, i_k\, \alpha_k}.$$

Then the same representation can be written in matrix-product form as

$$A_{i_1 i_2 \cdots i_d} = G^{(1)}(i_1)\, G^{(2)}(i_2) \cdots G^{(d)}(i_d),$$

where the right-hand side is a 1×1 matrix, identified with a scalar.

The vector $r = (r_0, r_1, \dots, r_d)$ is called the *TT-rank vector*, and the integers

$$r_1, \dots, r_{d-1}$$

are called the *TT-ranks* of the representation.

Remark 2.20.2. *If a tensor A is only approximated by a tensor $\widetilde{A}$ of the above form, then $\widetilde{A}$ is called a* TT approximation *of A. In that case one writes*

$$A = \widetilde{A} + E,$$

where E is the residual tensor.

Example 2.20.3 (A concrete tensor train decomposition)**.** Consider the third-order tensor

$$A \in \mathbb{R}^{2 \times 2 \times 2}$$

defined by the two frontal slices

$$A_{::1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad A_{::2} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Equivalently, its entries are

$$A_{111} = 1, \quad A_{121} = 0, \quad A_{211} = 0, \quad A_{221} = 1,$$

and

$$A_{112} = 0, \quad A_{122} = 1, \quad A_{212} = 1, \quad A_{222} = 0.$$

We show that A admits a tensor train decomposition with TT-rank vector

$$(r_0, r_1, r_2, r_3) = (1, 2, 2, 1).$$

Define the TT-cores

$$G^{(1)} \in \mathbb{R}^{1 \times 2 \times 2}, \qquad G^{(2)} \in \mathbb{R}^{2 \times 2 \times 2}, \qquad G^{(3)} \in \mathbb{R}^{2 \times 2 \times 1}$$

through their matrix slices as follows:

$$G^{(1)}(1) = \begin{pmatrix} 1 & 0 \end{pmatrix}, \qquad G^{(1)}(2) = \begin{pmatrix} 0 & 1 \end{pmatrix},$$

$$G^{(2)}(1) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad G^{(2)}(2) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

$$G^{(3)}(1) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \qquad G^{(3)}(2) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Then, for every $(i_1, i_2, i_3) \in \{1, 2\}^3$, the TT formula gives

$$A_{i_1 i_2 i_3} = G^{(1)}(i_1)\, G^{(2)}(i_2)\, G^{(3)}(i_3).$$

Let us verify several entries explicitly.

For $(i_1, i_2, i_3) = (1, 1, 1)$, we obtain

$$A_{111} = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 1.$$

For $(i_1, i_2, i_3) = (1, 2, 2)$, we obtain

$$A_{122} = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = 1.$$

For $(i_1, i_2, i_3) = (2, 1, 1)$, we obtain

$$A_{211} = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 0.$$

For $(i_1, i_2, i_3) = (2, 2, 1)$, we obtain

$$A_{221} = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 1.$$

By similar calculations, all remaining entries agree with the given tensor. Hence

$$A_{i_1 i_2 i_3} = G^{(1)}(i_1)\, G^{(2)}(i_2)\, G^{(3)}(i_3) \qquad (\forall\, i_1, i_2, i_3 \in \{1, 2\}),$$

so this is a valid tensor train decomposition of A.

2.21 Tree Tensor Network (TTN)

A Tree Tensor Network is a loop-free hierarchical tensor decomposition on a tree, efficiently representing many-body states through local tensors and virtual bonds capturing correlations [150, 151, 152, 153, 154]. Related concepts such as MERA (Multiscale Entanglement Renormalization Ansatz) [155, 156, 157, 158] are also known.

Definition 2.21.1 (Tree Tensor Network (TTN))**.** [150, 151, 152] Let $\tau = (V, E)$ be a finite rooted tree with root r. For each vertex $v \in V$, let $\mathcal{H}_v$ be a finite-dimensional complex vector space, called the *physical space* at v, and write

$$\dim \mathcal{H}_v = d_v.$$

For each non-root vertex $v \in V \setminus \{r\}$, let $p(v)$ denote the parent of v, let $\mathrm{Ch}(v)$ denote the set of children of v, and let

$$\mathcal{B}_v \cong \mathbb{C}^{D_v}$$

be a finite-dimensional *bond space* attached to the edge $(p(v), v)$.

A *tree tensor network* on τ is a family of local tensors

$$A^{(r)} \in \mathcal{H}_r \otimes \bigotimes_{u \in \mathrm{Ch}(r)} \mathcal{B}_u,$$

and, for each $v \in V \setminus \{r\}$,

$$A^{(v)} \in \mathcal{H}_v \otimes \mathcal{B}_v^* \otimes \bigotimes_{u \in \mathrm{Ch}(v)} \mathcal{B}_u.$$

The *TTN state represented by* $\{A^{(v)}\}_{v \in V}$ is the tensor

$$|\Psi_A\rangle := \mathrm{Contr}\left(\bigotimes_{v \in V} A^{(v)} \right) \in \bigotimes_{v \in V} \mathcal{H}_v,$$

obtained by contracting, for every non-root vertex v, the factor $\mathcal{B}_v$ appearing in $A^{(p(v))}$ with the factor $\mathcal{B}_v^*$ appearing in $A^{(v)}$ via the canonical pairing

$$\langle \cdot, \cdot \rangle_v : \mathcal{B}_v^* \times \mathcal{B}_v \to \mathbb{C}.$$

Equivalently, after choosing bases

$$\mathcal{H}_v = \mathrm{span}\{\, |i_v\rangle : 1 \le i_v \le d_v \,\}, \qquad \mathcal{B}_v = \mathrm{span}\{\, |\alpha_v\rangle : 1 \le \alpha_v \le D_v \,\},$$

the coefficients of $|\Psi_A\rangle$ are given by

$$|\Psi_A\rangle = \sum_{(i_v)_{v \in V}} \Psi_{(i_v)_{v \in V}} \bigotimes_{v \in V} |i_v\rangle,$$

where

$$\Psi_{(i_v)_{v \in V}} = \sum_{(\alpha_v)_{v \in V \setminus \{r\}}} A^{(r)}_{i_r,\, (\alpha_u)_{u \in \mathrm{Ch}(r)}} \prod_{v \in V \setminus \{r\}} A^{(v)}_{i_v,\, \alpha_v,\, (\alpha_u)_{u \in \mathrm{Ch}(v)}}.$$

Any tensor in $\bigotimes_{v \in V} \mathcal{H}_v$ that admits such a representation is called a *tree tensor network state.*

Example 2.21.2 (A concrete Tree Tensor Network)**.** Consider the rooted tree

$$\tau = (V, E), \qquad V = \{r, a, b\}, \qquad E = \{(r, a), (r, b)\},$$

where r is the root and a, b are its children. Thus,

$$\mathrm{Ch}(r) = \{a, b\}, \qquad \mathrm{Ch}(a) = \mathrm{Ch}(b) = \emptyset.$$

Let all physical spaces be qubit spaces:

$$\mathcal{H}_r = \mathcal{H}_a = \mathcal{H}_b = \mathbb{C}^2,$$

with standard basis

$$\{|0\rangle, |1\rangle\}.$$

Let the bond spaces be

$$\mathcal{B}_a = \mathcal{B}_b = \mathbb{C}^2,$$

with basis

$$\{|\alpha_0\rangle, |\alpha_1\rangle\},$$

and let

$$\{\langle\alpha_0|, \langle\alpha_1|\}$$

be the corresponding dual bases of $\mathcal{B}_a^*$ and $\mathcal{B}_b^*$.

Define the local tensors as follows:

$$A^{(r)} = |0\rangle_r \otimes |\alpha_0\rangle_a \otimes |\alpha_0\rangle_b + |1\rangle_r \otimes |\alpha_1\rangle_a \otimes |\alpha_1\rangle_b \in \mathcal{H}_r \otimes \mathcal{B}_a \otimes \mathcal{B}_b,$$

$$A^{(a)} = |0\rangle_a \otimes \langle\alpha_0| + |1\rangle_a \otimes \langle\alpha_1| \in \mathcal{H}_a \otimes \mathcal{B}_a^*,$$

and

$$A^{(b)} = |0\rangle_b \otimes \langle\alpha_0| + |1\rangle_b \otimes \langle\alpha_1| \in \mathcal{H}_b \otimes \mathcal{B}_b^*.$$

The TTN state is

$$|\Psi\rangle = \mathrm{Contr}\big(A^{(r)} \otimes A^{(a)} \otimes A^{(b)}\big) \in \mathcal{H}_r \otimes \mathcal{H}_a \otimes \mathcal{H}_b.$$

Using the canonical pairings

$$\langle\alpha_i, \alpha_j\rangle = \delta_{ij},$$

we obtain

$$|\Psi\rangle = |0\rangle_r \otimes |0\rangle_a \otimes |0\rangle_b + |1\rangle_r \otimes |1\rangle_a \otimes |1\rangle_b.$$

Hence

$$|\Psi\rangle = |000\rangle + |111\rangle,$$

which is the (unnormalized) three-qubit GHZ state.

Therefore, this gives a concrete example of a tree tensor network state on a binary rooted tree.

Proposition 2.21.3 (Well-definedness)**.** *The tensor $|\Psi_A\rangle$ is well-defined and does not depend on the order in which the virtual indices are contracted.*

Proof. Once bases are fixed, every coefficient $\Psi_{(i_v)_{v\in V}}$ is a finite sum of products of complex numbers over the virtual indices $(\alpha_v)_{v\in V\setminus\{r\}}$. Since multiplication in $\mathbb{C}$ is associative and finite sums may be reordered without changing their value, any contraction order yields the same coefficient. Hence the total contraction defines a unique tensor $|\Psi_A\rangle$. □

2.22 Projected Entangled Pair State (PEPS)

A PEPS is a tensor-network state built by projecting entangled virtual pairs onto physical lattice sites, efficiently encoding many-body correlations.

Definition 2.22.1 (Projected Entangled Pair State (PEPS))**.** Let $G = (V, E)$ be a finite undirected graph. For each vertex $v \in V$, let

$$\mathcal{H}_v \cong \mathbb{C}^d$$

be the physical Hilbert space at v, and let $D \geq 1$ be an integer, called the bond dimension.

For each edge $e = \{u, v\} \in E$, attach two virtual spaces

$$\mathbb{C}^D_{e,u} \quad \text{and} \quad \mathbb{C}^D_{e,v},$$

and define the maximally entangled vector

$$|\Omega_e\rangle := \sum_{\alpha=1}^{D} |\alpha\rangle_{e,u} \otimes |\alpha\rangle_{e,v} \in \mathbb{C}^D_{e,u} \otimes \mathbb{C}^D_{e,v}.$$

For each vertex $v \in V$, let

$$\mathcal{V}_v := \bigotimes_{e \ni v} \mathbb{C}^D_{e,v},$$

where the tensor product is taken over all edges incident to v, and choose a linear map

$$A_v : \mathcal{V}_v \to \mathcal{H}_v.$$

Then the vector

$$|\Psi\rangle := \left(\bigotimes_{v \in V} A_v\right) \left(\bigotimes_{e \in E} |\Omega_e\rangle\right) \in \bigotimes_{v \in V} \mathcal{H}_v$$

is called a *projected entangled pair state* (PEPS) on G.

In particular, when G is a one-dimensional chain, this construction reduces to a matrix product state (MPS).

Example 2.22.2 (A concrete PEPS on a square graph)**.** Let

$$G = (V, E), \qquad V = \{1, 2, 3, 4\}, \qquad E = \big\{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 1\}\big\},$$

so that G is a 4-cycle (a square plaquette).

For each vertex $v \in V$, let the physical space be

$$\mathcal{H}_v = \mathbb{C}^2$$

with standard basis

$$\{|0\rangle, |1\rangle\}.$$

Choose bond dimension $D = 2$. For each edge $e = \{u, v\} \in E$, let

$$|\Omega_e\rangle = |00\rangle_{e,u,e,v} + |11\rangle_{e,u,e,v} \in \mathbb{C}^2_{e,u} \otimes \mathbb{C}^2_{e,v}$$

be the maximally entangled virtual pair.

At each vertex, the virtual space is the tensor product of the two incident virtual spaces. Fix the following order:

$$\mathcal{V}_1 = \mathbb{C}^2_{12,1} \otimes \mathbb{C}^2_{41,1}, \qquad \mathcal{V}_2 = \mathbb{C}^2_{12,2} \otimes \mathbb{C}^2_{23,2},$$
$$\mathcal{V}_3 = \mathbb{C}^2_{23,3} \otimes \mathbb{C}^2_{34,3}, \qquad \mathcal{V}_4 = \mathbb{C}^2_{34,4} \otimes \mathbb{C}^2_{41,4}.$$

Define the local linear maps

$$A_v : \mathcal{V}_v \to \mathcal{H}_v \qquad (v = 1, 2, 3, 4)$$

by

$$A_v\big(|\alpha\rangle \otimes |\beta\rangle\big) = \delta_{\alpha\beta}\, |\alpha\rangle, \qquad \alpha, \beta \in \{0, 1\}.$$

Thus each A_v maps $|00\rangle \mapsto |0\rangle$, $|11\rangle \mapsto |1\rangle$, and annihilates $|01\rangle$ and $|10\rangle$.

The corresponding PEPS is

$$|\Psi\rangle = \left(\bigotimes_{v=1}^{4} A_v\right)\left(|\Omega_{12}\rangle \otimes |\Omega_{23}\rangle \otimes |\Omega_{34}\rangle \otimes |\Omega_{41}\rangle\right).$$

Writing

$$|\Omega_{ij}\rangle = \sum_{\alpha_{ij}=0}^{1} |\alpha_{ij}\rangle_{ij,i} \otimes |\alpha_{ij}\rangle_{ij,j},$$

we obtain

$$|\Psi\rangle = \sum_{\alpha_{12},\alpha_{23},\alpha_{34},\alpha_{41}\in\{0,1\}} \delta_{\alpha_{12},\alpha_{41}}\delta_{\alpha_{12},\alpha_{23}}\delta_{\alpha_{23},\alpha_{34}}\delta_{\alpha_{34},\alpha_{41}}\, |\alpha_{12}\rangle_1 \otimes |\alpha_{12}\rangle_2 \otimes |\alpha_{23}\rangle_3 \otimes |\alpha_{34}\rangle_4.$$

Hence only the assignments

$$\alpha_{12} = \alpha_{23} = \alpha_{34} = \alpha_{41} = 0 \quad \text{or} \quad \alpha_{12} = \alpha_{23} = \alpha_{34} = \alpha_{41} = 1$$

survive, and therefore

$$|\Psi\rangle = |0000\rangle + |1111\rangle.$$

Thus this construction gives a concrete PEPS on a square graph. After normalization, one obtains

$$\frac{1}{\sqrt{2}}\big(|0000\rangle + |1111\rangle\big).$$

2.23 Projected Entangled Simplex State (PESS)

A projected entangled simplex state is a tensor-network state obtained by placing entangled virtual tensors on simplices and projecting the incident virtual degrees of freedom onto physical site spaces [159, 160].

Definition 2.23.1 (Projected Entangled Simplex State (PESS))**.** Let L be a finite lattice, and let Σ be a family of simplices (higher-order cells) covering L. For each site $v \in L$, let

$$\mathcal{H}_v \cong \mathbb{C}^{d_v}$$

be the physical Hilbert space at v, and let $\mathcal{V}_{v,s} \cong \mathbb{C}^D$ be a virtual bond space associated with the incidence of v in a simplex $s \in \Sigma$.

For each simplex $s = \{v_1, \dots, v_m\} \in \Sigma$, choose an *entangled simplex tensor*

$$S^{(s)} \in \mathcal{V}_{v_1,s} \otimes \cdots \otimes \mathcal{V}_{v_m,s}.$$

For each site $v \in L$, choose a local *projection tensor*

$$A^{(v)} : \bigotimes_{s\ni v} \mathcal{V}_{v,s} \longrightarrow \mathcal{H}_v.$$

The vector

$$|\Psi\rangle := \mathrm{Contr}\left(\bigotimes_{s\in\Sigma} S^{(s)} \otimes \bigotimes_{v\in L} A^{(v)}\right) \in \bigotimes_{v\in L} \mathcal{H}_v$$

obtained by contracting all virtual indices is called a *projected entangled simplex state* (PESS).

Thus, a PESS is a tensor-network state in which multipartite entanglement is assigned to simplices rather than only to edges. In the special case where every simplex has two sites, a PESS reduces to a PEPS.

Example 2.23.2 (A concrete Projected Entangled Simplex State)**.** Let

$$L = \{v_1, v_2, v_3\}$$

be a finite lattice consisting of three sites, and let

$$\Sigma = \{s\}, \qquad s = \{v_1, v_2, v_3\},$$

so that Σ is a family of simplices covering L.

For each site $v_i \in L$, let the physical Hilbert space be

$$\mathcal{H}_{v_i} = \mathbb{C}^2$$

with standard basis

$$\{|0\rangle, |1\rangle\}.$$

Choose bond dimension $D = 2$, and for each incidence (v_i, s), let

$$\mathcal{V}_{v_i,s} \cong \mathbb{C}^2$$

with basis

$$\{|\alpha_0\rangle, |\alpha_1\rangle\}.$$

Define the entangled simplex tensor

$$S^{(s)} = |\alpha_0\rangle_{v_1,s} \otimes |\alpha_0\rangle_{v_2,s} \otimes |\alpha_0\rangle_{v_3,s} + |\alpha_1\rangle_{v_1,s} \otimes |\alpha_1\rangle_{v_2,s} \otimes |\alpha_1\rangle_{v_3,s} \in \mathcal{V}_{v_1,s} \otimes \mathcal{V}_{v_2,s} \otimes \mathcal{V}_{v_3,s}.$$

For each site $v_i \in L$, define the local projection tensor

$$A^{(v_i)} : \mathcal{V}_{v_i,s} \to \mathcal{H}_{v_i}$$

by

$$A^{(v_i)}(|\alpha_0\rangle) = |0\rangle, \qquad A^{(v_i)}(|\alpha_1\rangle) = |1\rangle.$$

Then the corresponding PESS is

$$|\Psi\rangle = \mathrm{Contr}\Big(S^{(s)} \otimes A^{(v_1)} \otimes A^{(v_2)} \otimes A^{(v_3)}\Big) \in \mathcal{H}_{v_1} \otimes \mathcal{H}_{v_2} \otimes \mathcal{H}_{v_3}.$$

Since each $A^{(v_i)}$ simply projects the virtual basis onto the physical basis, we obtain

$$|\Psi\rangle = |0\rangle_{v_1} \otimes |0\rangle_{v_2} \otimes |0\rangle_{v_3} + |1\rangle_{v_1} \otimes |1\rangle_{v_2} \otimes |1\rangle_{v_3}.$$

Hence

$$|\Psi\rangle = |000\rangle + |111\rangle,$$

which is the unnormalized three-qubit GHZ state.

Therefore, this gives a concrete example of a projected entangled simplex state.

2.24 MultiMeta-Graph

Unlike an ordinary metagraph, where each vertex is a single graph, a MultiMetaGraph allows each vertex to represent a finite nonempty family of graphs. Thus, one higher-level vertex may encode a module, portfolio, layer, cluster, or any other grouped unit consisting of several graphs.

Notation 2.24.1. For a set X, let $\mathcal{P}_{\mathrm{fin}}(X)$ denote the family of all finite subsets of X, and let

$$\mathcal{P}^*_{\mathrm{fin}}(X) := \mathcal{P}_{\mathrm{fin}}(X) \setminus \{\emptyset\}.$$

Definition 2.24.2 (MultiMetaGraph)**.** Fix a nonempty universe $\mathfrak{G}$ of finite graphs (undirected, loopless by default) and a nonempty family

$$\mathcal{R} \subseteq \mathcal{P}\Big(\mathcal{P}^*_{\mathrm{fin}}(\mathfrak{G}) \times \mathcal{P}^*_{\mathrm{fin}}(\mathfrak{G})\Big)$$

of admissible binary relations on finite nonempty families of graphs. A *MultiMetaGraph* over $(\mathfrak{G}, \mathcal{R})$ is a directed labelled multigraph

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

such that

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

and

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

Each element $v \in V$ is called a *multi-meta-vertex*; it is itself a finite nonempty family of graphs. If

$$v = \{G_1, \ldots, G_m\},$$

then we say that v *contains* the graphs $G_1, \ldots, G_m$. For $e \in E$ with $\lambda(e) = R$, we write

$$s(e) \xrightarrow{R} t(e)$$

and call e a *multi-meta-edge*.

Remark 2.24.3. *A MultiMetaGraph is still a graph-like object at the top level, but its vertices are no longer single graphs; they are finite graph-families. The same base graph may belong to several distinct multi-meta-vertices unless one explicitly imposes disjointness. If every relation in $\mathcal{R}$ is symmetric, then the MultiMetaGraph may be regarded as an undirected labelled multigraph.*

Example 2.24.4 (A concrete MultiMetaGraph of graph-families)**.** Let $\mathfrak{G}$ be a universe of finite simple graphs containing the following four graphs:

$$G_1 = P_2, \qquad G_2 = P_3, \qquad G_3 = K_3, \qquad G_4 = P_4,$$

where P_n denotes the path graph on n vertices and K_3 denotes the triangle graph.

Define two admissible binary relations on finite nonempty families of graphs:

$$R_{\mathrm{com}} := \Big\{(A, B) \in \mathcal{P}^*_{\mathrm{fin}}(\mathfrak{G}) \times \mathcal{P}^*_{\mathrm{fin}}(\mathfrak{G}) \;\Big|\; A \cap B \neq \emptyset\Big\},$$

and

$$R_{\mathrm{inc}} := \Big\{(A, B) \in \mathcal{P}^*_{\mathrm{fin}}(\mathfrak{G}) \times \mathcal{P}^*_{\mathrm{fin}}(\mathfrak{G}) \;\Big|\; \exists H \in A,\ \exists K \in B \text{ such that } |V(K)| = |V(H)| + 1\Big\}.$$

Let

$$\mathcal{R} = \{R_{\mathrm{com}}, R_{\mathrm{inc}}\}.$$

Now define three multi-meta-vertices by

$$M_1 = \{G_1, G_2\}, \qquad M_2 = \{G_2, G_3\}, \qquad M_3 = \{G_4\}.$$

Then

$$V = \{M_1, M_2, M_3\} \subseteq \mathcal{P}^*_{\mathrm{fin}}(\mathfrak{G}).$$

Next, define the multi-meta-edge set

$$E = \{e_1, e_2\},$$

together with source, target, and label maps by

$$s(e_1) = M_1, \qquad t(e_1) = M_2, \qquad \lambda(e_1) = R_{\mathrm{com}},$$

and

$$s(e_2) = M_2, \qquad t(e_2) = M_3, \qquad \lambda(e_2) = R_{\mathrm{inc}}.$$

We verify the incidence condition. Since

$$M_1 \cap M_2 = \{G_2\} \neq \emptyset,$$

we have

$$(M_1, M_2) \in R_{\mathrm{com}}.$$

Also, $G_3 \in M_2$ has 3 vertices and $G_4 \in M_3$ has 4 vertices, so

$$|V(G_4)| = |V(G_3)| + 1,$$

hence

$$(M_2, M_3) \in R_{\mathrm{inc}}.$$

Therefore,

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

Consequently,

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

is a MultiMetaGraph over $(\mathfrak{G}, \mathcal{R})$.

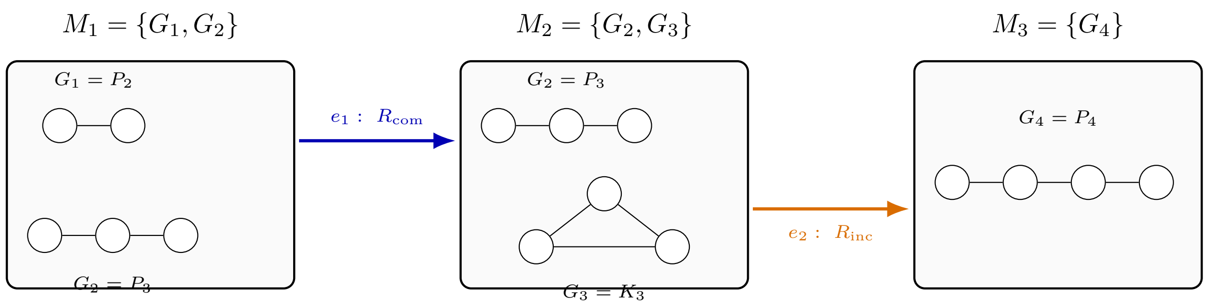
The multi-meta-vertex M_1 groups the two graphs P_2 and P_3, while M_2 groups P_3 and K_3. Thus the edge

$$M_1 \xrightarrow{R_{\mathrm{com}}} M_2$$

records that the two graph-families share a common graph object. The edge

$$M_2 \xrightarrow{R_{\mathrm{inc}}} M_3$$

records that a graph in M_3 has one more vertex than some graph in M_2. An illustration is given in Fig. 2.15.



$R_{com} : (A, B) \in R_{com} \iff A \cap B \neq \emptyset$ $R_{inc} : (A, B) \in R_{inc} \iff \exists H \in A, \exists K \in B, |V(K)| = |V(H)| + 1$

Here $M_1 \cap M_2 = \{G_2\}$, and $G_3 \in M_2$, $G_4 \in M_3$ satisfy $4 = 3 + 1$.

Figure 2.15: A concrete MultiMetaGraph. Each top-level vertex is a finite nonempty family of graphs, and each directed edge is labeled by a relation on graph-families.

2.25 Transfinite SuperHyperGraph

A transfinite superhypergraph models hierarchical set-based vertices with downward closure, connecting objects across infinitely many ordinally organized transfinite levels coherently.

$$\mathcal{P}^*(X) := \mathcal{P}(X) \setminus \{\emptyset\}, \qquad [V]^2 := \{\{x, y\} \subseteq V : x \neq y\}.$$

Definition 2.25.1 (Transfinite powerset hierarchy)**.** Let V_0 be a nonempty set of atoms, and let α be an ordinal. Define a family $(\mathcal{V}_\beta)_{\beta \leq \alpha}$ by transfinite recursion as follows:

$$\mathcal{V}_0 := V_0,$$

$$\mathcal{V}_{\beta+1} := \mathcal{P}^*(\mathcal{V}_\beta) \qquad (\beta < \alpha),$$

and, for every limit ordinal $\lambda \leq \alpha$,

$$\mathcal{V}_\lambda := \bigcup_{\beta < \lambda} \mathcal{V}_\beta.$$

The associated transfinite universe of height α is

$$\mathfrak{U}_\alpha(V_0) := \bigcup_{\beta \leq \alpha} \mathcal{V}_\beta.$$

For each $x \in \mathfrak{U}_\alpha(V_0)$, its *level* (or rank) is defined by

$$\ell(x) := \min\{\beta \leq \alpha : x \in \mathcal{V}_\beta\}.$$

Definition 2.25.2 (Transfinite SuperHyperGraph)**.** A *transfinite SuperHyperGraph of height* α on the base set V_0 is a pair

$$\mathcal{H}^{(\alpha)} = (V, E)$$

such that

1. $$\emptyset \neq V \subseteq \mathfrak{U}_\alpha(V_0);$$

2. for every $X \in V \setminus V_0$,
$$X \subseteq V;$$
that is, V is downward closed under the membership relation;

3. $$E \subseteq \mathcal{P}^*(V).$$

The elements of V are called *transfinite supervertices*, and the elements of E are called *transfinite superhyperedges*.

Example 2.25.3 (A concrete Transfinite SuperHyperGraph of height $\omega + 1$)**.** Let the base set be

$$V_0 = \{a, b\},$$

and let

$$(\mathcal{V}_\beta)_{\beta \le \omega+1}$$

be the transfinite powerset hierarchy from the preceding definition.

First define several lower-level objects:

$$s := \{a\}, \qquad u := \{a, b\} \in \mathcal{V}_1,$$

and

$$w := \{s\} = \{\{a\}\} \in \mathcal{V}_2.$$

Since

$$\mathcal{V}_\omega = \bigcup_{n<\omega} \mathcal{V}_n,$$

we have

$$a, b, s, u, w \in \mathcal{V}_\omega.$$

Hence the mixed-level sets

$$p := \{a, w\}, \qquad q := \{b, u\}$$

belong to

$$\mathcal{V}_{\omega+1} = \mathcal{P}^*(\mathcal{V}_\omega).$$

Moreover, p and q do not belong to any finite level $\mathcal{V}_n$ $(n < \omega)$, because each of them contains constituents coming from different finite ranks. Therefore

$$\ell(p) = \ell(q) = \omega + 1.$$

Now define the transfinite supervertex set by

$$V = \{a, b, s, u, w, p, q\} \subseteq \mathfrak{U}_{\omega+1}(V_0),$$

and define the transfinite superhyperedge family by

$$E = \big\{e_1 = \{s, u\},\ e_2 = \{w, p\},\ e_3 = \{p, q\}\big\} \subseteq \mathcal{P}^*(V).$$

Then

$$\mathcal{H}^{(\omega+1)} = (V, E)$$

is a Transfinite SuperHyperGraph of height $\omega + 1$. Indeed:

1. $V \subseteq \mathfrak{U}_{\omega+1}(V_0)$;
2. V is downward closed under membership, since

$$s = \{a\} \subseteq V, \qquad u = \{a, b\} \subseteq V, \qquad w = \{s\} \subseteq V,$$

$$p = \{a, w\} \subseteq V, \qquad q = \{b, u\} \subseteq V;$$

3. each e_i is a nonempty subset of V.

The hyperedge e_1 connects two finite-level supervertices, e_2 is a cross-level hyperedge joining a level-2 object and an $(\omega + 1)$-level object, and e_3 connects two genuinely transfinite supervertices. An illustration is given in Fig. 2.16.

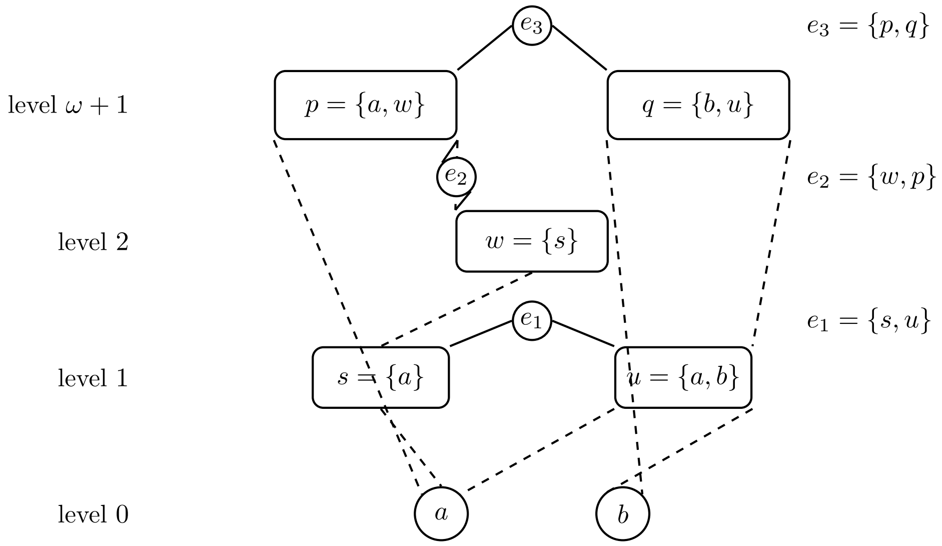


Figure 2.16: A concrete Transfinite SuperHyperGraph of height $\omega + 1$. Dashed lines indicate membership/containment relations used to witness downward closure, while e_1, e_2, e_3 denote transfinite superhyperedges.

2.26 Multi-Axis SuperHyperGraph

A Multi Axis SuperHyperGraph represents higher order relations among vertices carrying multi indexed iterated powerset coordinates, with cross level inclusion and hierarchical multidimensional organization rules.

Definition 2.26.1 (Multi-indexed iterated powerset)**.** Let $d \in \mathbb{N}$, and let $\mathbf{U} = (U_1, \ldots, U_d)$ be a d-tuple of finite nonempty sets. For each axis $r \in \{1, \ldots, d\}$, define

$$\mathcal{P}^0(U_r) := U_r, \qquad \mathcal{P}^{n+1}(U_r) := \mathcal{P}(\mathcal{P}^n(U_r)) \quad (n \in \mathbb{N}_0).$$

For a multi-index $\mathbf{n} = (n_1, \ldots, n_d) \in \mathbb{N}_0^d$, define the $\mathbf{n}$-layer by

$$\mathbb{P}^{\mathbf{n}}(\mathbf{U}) := \prod_{r=1}^{d} \mathcal{P}^{n_r}(U_r).$$

An element $x \in \mathbb{P}^{\mathbf{n}}(\mathbf{U})$ is written as $x = (x_1, \ldots, x_d)$ with $x_r \in \mathcal{P}^{n_r}(U_r)$.

Example 2.26.2 (Multi-indexed iterated powerset in two axes)**.** Let

$$d = 2, \qquad U_1 = \{a, b\}, \qquad U_2 = \{1, 2, 3\},$$

so the base system is

$$\mathbf{U} = (U_1, U_2).$$

Consider the multi-index

$$\mathbf{n} = (1, 2).$$

Then

$$\mathbb{P}^{(1,2)}(\mathbf{U}) = \mathcal{P}^1(U_1) \times \mathcal{P}^2(U_2) = \mathcal{P}(U_1) \times \mathcal{P}(\mathcal{P}(U_2)).$$

Hence an element of $\mathbb{P}^{(1,2)}(\mathbf{U})$ has the form

$$x = (x_1, x_2) \quad \text{with} \quad x_1 \subseteq U_1, \quad x_2 \subseteq \mathcal{P}(U_2).$$

For instance, define

$$x := \big(\{a\}, \ \{\{1\}, \{1, 3\}\}\big) \in \mathbb{P}^{(1,2)}(\mathbf{U}).$$

Its first coordinate is a subset of U_1, and its second coordinate is a set of subsets of U_2.
Now apply the axis-wise singleton lift along the first axis:

$$\Sigma_1^{(1,2)}(x) = \big(\{\{a\}\}, \ \{\{1\}, \{1, 3\}\}\big) \in \mathbb{P}^{(2,2)}(\mathbf{U}).$$

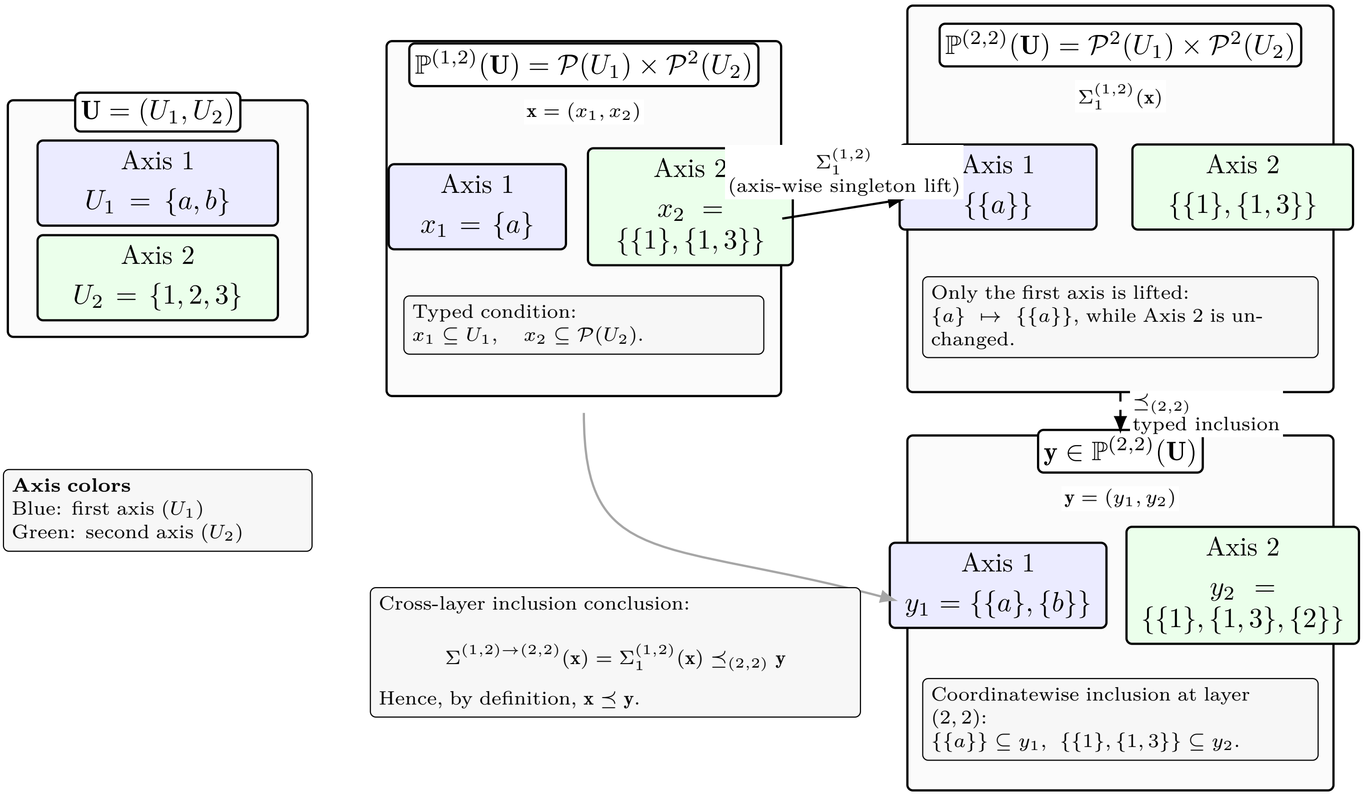


Figure 2.17: A two-axis multi-indexed iterated powerset example (Example 2.26.10). The figure shows the element $\mathbf{x} \in \mathbb{P}^{(1,2)}(\mathbf{U})$, the axis-wise singleton lift $\Sigma_1^{(1,2)}(\mathbf{x}) \in \mathbb{P}^{(2,2)}(\mathbf{U})$, and the typed coordinatewise inclusion into $\mathbf{y} \in \mathbb{P}^{(2,2)}(\mathbf{U})$.

Indeed, the first coordinate has been lifted from $\mathcal{P}(U_1)$ to $\mathcal{P}(\mathcal{P}(U_1))$, while the second coordinate is unchanged.

Let

$$y := \big(\{\{a\}, \{b\}\},\ \{\{1\}, \{1, 3\}, \{2\}\}\big) \in \mathbb{P}^{(2,2)}(\mathbf{U}).$$

Then

$$\Sigma^{(1,2)\to(2,2)}(x) = \Sigma_1^{(1,2)}(x) \preceq_{(2,2)} y,$$

because

$$\{\{a\}\} \subseteq \{\{a\}, \{b\}\} \quad \text{and} \quad \{\{1\}, \{1, 3\}\} \subseteq \{\{1\}, \{1, 3\}, \{2\}\}.$$

Therefore, by the cross-layer inclusion definition, we have

$$x \preceq y.$$

This example illustrates how different axes can be lifted independently and then compared via typed coordinatewise inclusion. An overview diagram of this example is provided in Fig. 2.17.

Definition 2.26.3 (Axis-wise singleton lift)**.** Let $\mathbf{e}_r$ denote the r-th standard basis vector of $\mathbb{N}_0^d$. For $\mathbf{n} \in \mathbb{N}_0^d$, define

$$\Sigma_r^{\mathbf{n}} : \mathbb{P}^{\mathbf{n}}(\mathbf{U}) \to \mathbb{P}^{\mathbf{n}+\mathbf{e}_r}(\mathbf{U})$$

by

$$\Sigma_r^{\mathbf{n}}(x_1, \ldots, x_d) := (x_1, \ldots, x_{r-1}, \{x_r\}, x_{r+1}, \ldots, x_d).$$

If $\mathbf{n} \leq \mathbf{m}$ coordinatewise, define

$$\Sigma^{\mathbf{n}\to\mathbf{m}} : \mathbb{P}^{\mathbf{n}}(\mathbf{U}) \to \mathbb{P}^{\mathbf{m}}(\mathbf{U})$$

as the iterated composition of the axis-wise singleton lifts, applied $(m_r - n_r)$ times along axis r for each r.

Definition 2.26.4 (Typed coordinatewise inclusion)**.** Fix $\mathbf{n} = (n_1, \ldots, n_d) \in \mathbb{N}_0^d$. For each axis r, define a relation $\preceq_r^{(n_r)}$ on $\mathcal{P}^{n_r}(U_r)$ by

$$a \preceq_r^{(n_r)} b \iff \begin{cases} a = b, & n_r = 0, \\ a \subseteq b, & n_r \geq 1. \end{cases}$$

Then define $\preceq_{\mathbf{n}}$ on $\mathbb{P}^{\mathbf{n}}(\mathbf{U})$ by

$$x \preceq_{\mathbf{n}} y \iff x_r \preceq_r^{(n_r)} y_r \quad (\forall r = 1, \ldots, d).$$

Definition 2.26.5 (Cross-layer inclusion)**.** Let $x \in \mathbb{P}^{\mathbf{n}}(\mathbf{U})$ and $y \in \mathbb{P}^{\mathbf{m}}(\mathbf{U})$. If $\mathbf{n} \le \mathbf{m}$, define

$$x \preceq y \iff \Sigma^{\mathbf{n}\to\mathbf{m}}(x) \preceq_{\mathbf{m}} y.$$

If $\mathbf{n} \not\le \mathbf{m}$, we regard x and y as incomparable (with respect to this relation).

Definition 2.26.6 (Typed graded multi-axis vertex universe)**.** Let $\mathbf{U} = (U_1, \dots, U_d)$ be a d-axis base system, and let $I \subseteq \mathbb{N}_0^d$ be a finite index set of admissible multi-levels. Define

$$\mathfrak{V}_I(\mathbf{U}) := \bigsqcup_{\mathbf{n}\in I} \big(\{\mathbf{n}\} \times \mathbb{P}^{\mathbf{n}}(\mathbf{U})\big).$$

An element of $\mathfrak{V}_I(\mathbf{U})$ is written $(\mathbf{n}, x)$ with $x \in \mathbb{P}^{\mathbf{n}}(\mathbf{U})$.

Definition 2.26.7 (Multi-Axis SuperHyperGraph)**.** A *Multi-Axis SuperHyperGraph* (MASHG) on $(\mathbf{U}, I)$ is a pair

$$\mathcal{H} = (V, E),$$

where

$$V \subseteq \mathfrak{V}_I(\mathbf{U})$$

is a finite set of vertices (multi-axis supervertices), and

$$E \subseteq \mathcal{P}(V) \setminus \{\varnothing\}$$

is a finite set of hyperedges (multi-axis superhyperedges).

Definition 2.26.8 (Level map and content map)**.** Let $\mathcal{H} = (V, E)$ be a MASHG. For each vertex $v = (\mathbf{n}, x) \in V$, define

$$\lambda(v) := \mathbf{n}, \qquad \chi(v) := x.$$

Thus $\lambda : V \to I$ is the level map, and $\chi(v) \in \mathbb{P}^{\lambda(v)}(\mathbf{U})$ is the typed content of v.

Definition 2.26.9 (Multi-axis inclusion between vertices)**.** Let $u, v \in V$. We write $u \preceq_{\mathcal{H}} v$ if

$$\lambda(u) \le \lambda(v) \quad \text{and} \quad \Sigma^{\lambda(u)\to\lambda(v)}(\chi(u)) \preceq_{\lambda(v)} \chi(v).$$

Example 2.26.10 (Multi-indexed iterated powerset in two axes)**.** Let

$$d = 2, \qquad U_1 = \{a, b\}, \qquad U_2 = \{1, 2, 3\},$$

so the base system is

$$\mathbf{U} = (U_1, U_2).$$

Consider the multi-index

$$\mathbf{n} = (1, 2).$$

Then

$$\mathbb{P}^{(1,2)}(\mathbf{U}) = \mathcal{P}^1(U_1) \times \mathcal{P}^2(U_2) = \mathcal{P}(U_1) \times \mathcal{P}(\mathcal{P}(U_2)).$$

Hence an element of $\mathbb{P}^{(1,2)}(\mathbf{U})$ has the form

$$x = (x_1, x_2) \quad \text{with} \quad x_1 \subseteq U_1, \quad x_2 \subseteq \mathcal{P}(U_2).$$

For instance, define

$$x := \big(\{a\},\ \{\{1\}, \{1,3\}\}\big) \in \mathbb{P}^{(1,2)}(\mathbf{U}).$$

Its first coordinate is a subset of U_1, and its second coordinate is a set of subsets of U_2.

Now apply the axis-wise singleton lift along the first axis:

$$\Sigma_1^{(1,2)}(x) = \big(\{\{a\}\},\ \{\{1\}, \{1,3\}\}\big) \in \mathbb{P}^{(2,2)}(\mathbf{U}).$$

Indeed, the first coordinate has been lifted from $\mathcal{P}(U_1)$ to $\mathcal{P}(\mathcal{P}(U_1))$, while the second coordinate is unchanged.

Let

$$y := \big(\{\{a\}, \{b\}\},\ \{\{1\}, \{1,3\}, \{2\}\}\big) \in \mathbb{P}^{(2,2)}(\mathbf{U}).$$

Then

$$\Sigma^{(1,2)\to(2,2)}(x) = \Sigma_1^{(1,2)}(x) \preceq_{(2,2)} y,$$

because

$$\{\{a\}\} \subseteq \{\{a\}, \{b\}\} \quad \text{and} \quad \{\{1\}, \{1,3\}\} \subseteq \{\{1\}, \{1,3\}, \{2\}\}.$$

Therefore, by the cross-layer inclusion definition, we have

$$x \preceq y.$$

This example illustrates how different axes can be lifted independently and then compared via typed coordinatewise inclusion.

2.27 Iterated Multi-Edge Graph

An iterated multi-edge graph keeps ordinary vertices but represents edges as an r-fold iterated multiset of endpoint-pairs, capturing nested multiplicities and grouped edge structure.

Definition 2.27.1 (Finite multiset and iterated multiset)**.** Let X be a set. A *finite multiset* on X is a function

$$m : X \to \mathbb{N}_0$$

with finite support $\operatorname{supp}(m) := \{x \in X \mid m(x) > 0\}$. Let $\mathsf{M}(X)$ denote the set of all finite multisets on X. For $r \in \mathbb{N}_0$, define iterated multiset universes by

$$\mathsf{M}^0(X) := X, \qquad \mathsf{M}^{r+1}(X) := \mathsf{M}\big(\mathsf{M}^r(X)\big).$$

Definition 2.27.2 (Base edge-type universe)**.** Let V be a finite set. Define the set of *unordered pairs with repetition* by

$$\binom{V}{2}^m := \{\{\{u, v\}\} \mid u, v \in V\},$$

where $\{\{u, v\}\}$ denotes the 2-element multiset with endpoints u, v (so $\{\{v, v\}\}$ represents a loop at v).

Definition 2.27.3 (Iterated Multi-Edge Graph of order r)**.** Let V be a finite vertex set and let $r \in \mathbb{N}$. An *Iterated Multi-Edge Graph of order* r is a pair

$$\mathsf{IMEG}^{(r)} = (V, \mathcal{E}^{(r)}),$$

where the *edge structure* is an r-fold iterated multiset over the base edge-type universe:

$$\mathcal{E}^{(r)} \in \mathsf{M}^r\Big(\binom{V}{2}^m\Big).$$

Thus the collection of edges is not merely a multiset of endpoint-pairs (order 1), but an iterated multiset of such data (order r), allowing nested multiplicity/grouping structures among edges.

Remark 2.27.4 (Order 1 recovers ordinary multigraphs)**.** *When $r = 1$, we have $\mathcal{E}^{(1)} \in \mathsf{M}(\binom{V}{2}^m)$, which is exactly an undirected multigraph edge multiset (equivalently, an edge-multiplicity map $\binom{V}{2}^m \to \mathbb{N}_0$).*

Example 2.27.5 (An Iterated Multi-Edge Graph of order 2 (grouped multiplicities))**.** Let

$$V = \{1, 2, 3\}$$

be the vertex set. Then the base edge-type universe is

$$\binom{V}{2}^m = \{\{\{1,1\}\}, \{\{2,2\}\}, \{\{3,3\}\}, \{\{1,2\}\}, \{\{1,3\}\}, \{\{2,3\}\}\},$$

where $\{\{i, i\}\}$ represents a loop-type pair at i.

First-level edge multisets. Define two ordinary edge-multisets (elements of $\mathsf{M}(\binom{V}{2}^m)$) by specifying their nonzero multiplicities:

$$m_A(\{\{1,2\}\}) = 2, \qquad m_A(\{\{2,3\}\}) = 1,$$

and $m_A(e) = 0$ for all other $e \in \binom{V}{2}^m$; and

$$m_B(\{\{1,2\}\}) = 1, \qquad m_B(\{\{1,3\}\}) = 3,$$

and $m_B(e) = 0$ otherwise. Intuitively, m_A encodes "two parallel edges between 1 and 2 plus one edge between 2 and 3," while m_B encodes "one edge between 1 and 2 plus three parallel edges between 1 and 3."

Second-level (iterated) edge object. Now define an element of $\mathsf{M}^2(\binom{V}{2}^m) = \mathsf{M}(\mathsf{M}(\binom{V}{2}^m))$ by taking a finite multiset of these first-level multisets:

$$\mathcal{E}^{(2)} \in \mathsf{M}\Big(\mathsf{M}\Big(\binom{V}{2}^m\Big)\Big), \qquad \mathcal{E}^{(2)}(m_A) = 2, \quad \mathcal{E}^{(2)}(m_B) = 1,$$

and $\mathcal{E}^{(2)}(m) = 0$ for all other $m \in \mathsf{M}(\binom{V}{2}^m)$. Thus $\mathcal{E}^{(2)}$ consists of two "copies" of the edge multiset m_A and one "copy" of m_B, representing a grouped (nested) multiplicity structure on edges.

Then

$$\mathsf{IMEG}^{(2)} = \big(V, \mathcal{E}^{(2)}\big)$$

is an Iterated Multi-Edge Graph of order 2 in the sense of Definition 2.27.3. If one flattens $\mathcal{E}^{(2)}$ to an ordinary edge multiset (by summing multiplicities), the total counts become:

$$\{\{1,2\}\} : 2 \cdot 2 + 1 \cdot 1 = 5, \qquad \{\{2,3\}\} : 2 \cdot 1 = 2, \qquad \{\{1,3\}\} : 1 \cdot 3 = 3,$$

illustrating how the second-level structure aggregates multiple first-level edge configurations.

Theorem 2.27.6 (Well-definedness of Iterated Multi-Edge Graphs)**.** *Let V be a finite set and let $r \in \mathbb{N}$. Assume that $\binom{V}{2}^m$ denotes a fixed set of base edge-types over V (for example, $\binom{V}{2}$ itself, or a labeled copy of $\binom{V}{2}$ if one wishes to distinguish edge-types). Then the r-fold iterated multiset space*

$$\mathsf{M}^r\Big(\binom{V}{2}^m\Big)$$

is a well-defined set. Consequently, for every

$$\mathcal{E}^{(r)} \in \mathsf{M}^r\Big(\binom{V}{2}^m\Big),$$

the pair

$$\mathsf{IMEG}^{(r)} = (V, \mathcal{E}^{(r)})$$

is a well-defined mathematical object in the sense of Definition 2.27.3.

Proof. We use the standard finite-multiset construction

$$\mathsf{M}(X) = \{\mu : X \to \mathbb{N}_0 \mid \operatorname{supp}(\mu) \text{ is finite}\},$$

where $\operatorname{supp}(\mu) = \{x \in X : \mu(x) \neq 0\}$, and define iterates recursively by

$$\mathsf{M}^1(X) = \mathsf{M}(X), \qquad \mathsf{M}^{k+1}(X) = \mathsf{M}(\mathsf{M}^k(X)).$$

Set

$$X_0 := \binom{V}{2}^m.$$

By assumption, X_0 is a set. (Since V is finite, $\binom{V}{2}$ is finite; any fixed edge-type refinement of it is also a set.)

We now prove by induction on $r \geq 1$ that $\mathsf{M}^r(X_0)$ is a set.

Base case ($r = 1$). By definition,

$$\mathsf{M}^1(X_0) = \mathsf{M}(X_0)$$

is the set of finitely supported functions $X_0 \to \mathbb{N}_0$. Hence $\mathsf{M}^1(X_0)$ is well-defined.

Inductive step. Assume $\mathsf{M}^r(X_0)$ is a well-defined set for some $r \geq 1$. Then

$$\mathsf{M}^{r+1}(X_0) = \mathsf{M}\big(\mathsf{M}^r(X_0)\big)$$

is, again by the same finite-multiset construction, the set of finitely supported functions

$$\mathsf{M}^r(X_0) \to \mathbb{N}_0.$$

Therefore $\mathsf{M}^{r+1}(X_0)$ is also a well-defined set.

By induction, $\mathsf{M}^r(X_0)$ is well-defined for every $r \in \mathbb{N}$. Thus any choice $\mathcal{E}^{(r)} \in \mathsf{M}^r(X_0)$ determines a legitimate pair $(V, \mathcal{E}^{(r)})$, i.e., an Iterated Multi-Edge Graph of order r. □

2.28 Iterated Multi-Recursive Graph

An iterated multi-recursive graph uses iterated multisets for both vertices and edges: vertices are nested multisets, and edges are nested multisets of endpoint-pairs.

Definition 2.28.1 ((Recall) Finite multiset and iterated multiset)**.** Let X be a set. A *finite multiset* on X is a function

$$m : X \to \mathbb{N}_0$$

with finite support

$$\mathrm{supp}(m) := \{x \in X \mid m(x) > 0\}.$$

Let $\mathsf{M}(X)$ denote the set of all finite multisets on X. For $r \in \mathbb{N}_0$, define the *r-fold iterated multiset universes* recursively by

$$\mathsf{M}^0(X) := X, \qquad \mathsf{M}^{r+1}(X) := \mathsf{M}\big(\mathsf{M}^r(X)\big).$$

Definition 2.28.2 (Unordered pairs with repetition)**.** Let U be a set. Define

$$\binom{U}{2}^{m} := \{\{\{u, v\}\} \mid u, v \in U\},$$

the set of *unordered pairs with repetition* (i.e. 2-element multisets). Thus $\{\{u, u\}\}$ represents a loop-type pair at u.

Definition 2.28.3 (Iterated Multi-Recursive Graph of type (p, q))**.** Let X be a nonempty base set and let $p, q \in \mathbb{N}_0$. An *Iterated Multi-Recursive Graph of type* (p, q) *over* X is a pair

$$\mathsf{IMRG}^{(p,q)} = (V^{(p)}, E^{(q)})$$

satisfying:

1. (**Iterated multiset vertex object**)
$$V^{(p)} \in \mathsf{M}^p(X).$$
Let
$$\underline{V} := \mathrm{supp}(V^{(p)}) \subseteq \mathsf{M}^p(X)$$
be the underlying *set* of distinct vertex-objects.

2. (**Iterated multiset edge object**)
$$E^{(q)} \in \mathsf{M}^q\Big(\binom{\underline{V}}{2}^{m}\Big).$$
Hence the edge data is a q-fold iterated multiset of unordered endpoint-pairs (with repetition) drawn from the vertex set $\underline{V}$.

Elements of $\underline{V}$ are called *vertices*, and elements of $\mathrm{supp}(E^{(q)})$ may be viewed as *edge-types* (possibly nested, when $q \geq 2$) whose bottom-level atoms are unordered endpoint-pairs in $\binom{\underline{V}}{2}^{m}$.

Remark 2.28.4 (Special cases)**.**

- *If $p = 0$ and $q = 1$, then $V^{(0)} \in X$ is an ordinary vertex element and the above definition reduces (after choosing a vertex set) to an ordinary undirected multigraph (edges form a multiset of endpoint-pairs).*
- *If $p = n$ and $q = 1$, then $\mathsf{IMRG}^{(n,1)}$ is an* Iterated MultiGraph of order *n (vertically iterated multiset vertices, ordinary multiset edges).*
- *If $p = 0$ and $q = r$, then $\mathsf{IMRG}^{(0,r)}$ matches an* Iterated Multi-Edge Graph of order *r (ordinary vertices, iterated multiset edge structure).*

Example 2.28.5 (An Iterated Multi-Recursive Graph of type $(1, 2)$)**.** Let the base set be

$$X = \{a, b\}.$$

Take $p = 1$, so vertices are 1-fold multisets on X (elements of $\mathsf{M}(X)$). Define the vertex multiset

$$V^{(1)} \in \mathsf{M}(X)$$

by specifying its multiplicities:

$$V^{(1)}(a) = 2, \qquad V^{(1)}(b) = 1.$$

Thus $\operatorname{supp}(V^{(1)}) = \underline{V} = \{a, b\}$, so the underlying set of distinct vertex-objects is

$$\underline{V} = \{a, b\} \subseteq \mathsf{M}^1(X).$$

(Here we identify the elements $a, b \in X$ with the singleton multisets $\{\{a\}\}, \{\{b\}\} \in \mathsf{M}(X)$, so that $\underline{V}$ may be viewed as a subset of $\mathsf{M}^1(X)$.)

Now take $q = 2$, so the edge structure is a 2-fold iterated multiset on the endpoint-pair universe $\binom{V}{2}^m$. First compute

$$\binom{\underline{V}}{2}^m = \{\{\{a, a\}\}, \{\{b, b\}\}, \{\{a, b\}\}\}.$$

First-level edge multisets. Define two elements of $\mathsf{M}\big(\binom{V}{2}^m\big)$ by their nonzero multiplicities:

$$m_1(\{\{a, b\}\}) = 2, \qquad m_1(\{\{a, a\}\}) = 1,$$

(all other pairs have multiplicity 0), and

$$m_2(\{\{a, b\}\}) = 1, \qquad m_2(\{\{b, b\}\}) = 3,$$

(all other pairs have multiplicity 0). Intuitively, m_1 represents two parallel edges between a and b together with one loop at a, while m_2 represents one edge between a and b together with three loops at b.

Second-level (iterated) edge object. Now define

$$E^{(2)} \in \mathsf{M}^2\Big(\binom{\underline{V}}{2}^m\Big) = \mathsf{M}\Big(\mathsf{M}\big(\binom{\underline{V}}{2}^m\big)\Big)$$

by taking a multiset of the first-level edge multisets:

$$E^{(2)}(m_1) = 1, \qquad E^{(2)}(m_2) = 2,$$

and $E^{(2)}(m) = 0$ for all other $m \in \mathsf{M}\big(\binom{V}{2}^m\big)$. Thus $E^{(2)}$ contains one "copy" of the configuration m_1 and two "copies" of m_2.

Therefore,

$$\mathsf{IMRG}^{(1,2)} = \big(V^{(1)}, E^{(2)}\big)$$

is an Iterated Multi-Recursive Graph of type $(p, q) = (1, 2)$ over X in the sense of Definition 2.28.3. Here both the vertex object $V^{(1)}$ and the edge object $E^{(2)}$ carry nested multiset structure.

Lemma 2.28.6 (Well-definedness of finite multisets)**.** *Let X be a set and let $\mathsf{M}(X)$ denote the set of all finite multisets on X, i.e.*

$$\mathsf{M}(X) = \big\{\, m : X \to \mathbb{N}_0 \;\big|\; \operatorname{supp}(m) \textit{ is finite} \big\}, \qquad \operatorname{supp}(m) := \{x \in X \mid m(x) > 0\}.$$

Then:

1. *$\mathsf{M}(X)$ is a set.*
2. *For every $m \in \mathsf{M}(X)$, the support $\operatorname{supp}(m)$ is a finite subset of X.*

Proof. (1) Since $\mathsf{M}(X) \subseteq \mathbb{N}_0^X$ and $\mathbb{N}_0^X$ is a set, $\mathsf{M}(X)$ is a set.

(2) This holds by the defining condition "$\operatorname{supp}(m)$ is finite" for $m \in \mathsf{M}(X)$. ☐

Lemma 2.28.7 (Iterated multiset universes are sets, and supports are finite)**.** *Let X be a set and define $\mathsf{M}^0(X) := X$ and $\mathsf{M}^{r+1}(X) := \mathsf{M}(\mathsf{M}^r(X))$ for $r \geq 0$. Then for every $r \in \mathbb{N}$:*

1. *$\mathsf{M}^r(X)$ is a set.*
2. *Every element $M \in \mathsf{M}^r(X)$ (which is a multiset on $\mathsf{M}^{r-1}(X)$) has finite support $\operatorname{supp}(M) \subseteq \mathsf{M}^{r-1}(X)$.*

Proof. We prove (1)–(2) by induction on $r \geq 1$.

Base $r = 1$. We have $\mathsf{M}^1(X) = \mathsf{M}(X)$, which is a set by Lemma 2.28.6(1), and every $M \in \mathsf{M}(X)$ has finite support by Lemma 2.28.6(2).

Inductive step. Assume $\mathsf{M}^r(X)$ is a set. Then

$$\mathsf{M}^{r+1}(X) = \mathsf{M}(\mathsf{M}^r(X))$$

is a set by Lemma 2.28.6(1) applied to $X := \mathsf{M}^r(X)$. Moreover, any $M \in \mathsf{M}^{r+1}(X)$ is a finite multiset on $\mathsf{M}^r(X)$, hence $\mathrm{supp}(M) \subseteq \mathsf{M}^r(X)$ is finite by Lemma 2.28.6(2). □

Theorem 2.28.8 (Well-definedness of $\mathsf{IMRG}^{(p,q)}$)**.** *Let X be a nonempty set and let $p, q \in \mathbb{N}_0$. Let $V^{(p)} \in \mathsf{M}^p(X)$ be a vertex object and define the underlying vertex set by*

$$\underline{V} := \begin{cases} \{V^{(0)}\}, & p = 0, \\ \mathrm{supp}(V^{(p)}), & p \geq 1, \end{cases}$$

so that $\underline{V}$ is a set of distinct vertex-objects. Then:

1. *$\underline{V}$ is a finite set.*
2. *The unordered-pair-with-repetition universe $\binom{\underline{V}}{2}^m$ is a set.*
3. *The iterated edge universe $\mathsf{M}^q\big(\binom{\underline{V}}{2}^m\big)$ is a set.*
4. *Consequently, any choice of*

$$E^{(q)} \in \mathsf{M}^q\Big(\binom{\underline{V}}{2}^m\Big)$$

produces a well-defined Iterated Multi-Recursive Graph of type (p, q)

$$\mathsf{IMRG}^{(p,q)} = (V^{(p)}, E^{(q)})$$

in the sense of Definition 2.28.3.

Proof. (1) If $p = 0$, then $\underline{V} = \{V^{(0)}\}$ is a singleton, hence finite. If $p \geq 1$, then $V^{(p)} \in \mathsf{M}^p(X) = \mathsf{M}(\mathsf{M}^{p-1}(X))$, so $V^{(p)}$ is a finite multiset on $\mathsf{M}^{p-1}(X)$; therefore $\underline{V} = \mathrm{supp}(V^{(p)})$ is finite by Lemma 2.28.7(2).

(2) Since $\underline{V}$ is a set, the collection of 2-element multisets on $\underline{V}$, namely $\binom{\underline{V}}{2}^m = \{\{\{u, v\}\} \mid u, v \in \underline{V}\}$, is a set (it is a definable image of $\underline{V} \times \underline{V}$).

(3) By Lemma 2.28.7(1) applied to the set $\binom{\underline{V}}{2}^m$, the iterated multiset universe $\mathsf{M}^q\big(\binom{\underline{V}}{2}^m\big)$ is a set.

(4) With (1)–(3), the expression $E^{(q)} \in \mathsf{M}^q\big(\binom{\underline{V}}{2}^m\big)$ is meaningful, and hence the pair $(V^{(p)}, E^{(q)})$ satisfies the two requirements in Definition 2.28.3: the vertex object is an iterated multiset of depth p over X, and the edge object is an iterated multiset of depth q over the endpoint-pair universe derived from the underlying vertex set $\underline{V}$. Therefore $\mathsf{IMRG}^{(p,q)}$ is well-defined. □

2.29 HyperMatroid

A hypermatroid encodes matroid dependence via circuit hyperedges: minimal dependent subsets satisfying elimination, determining independence combinatorially.

Definition 2.29.1 (HyperMatroid (circuit-hypergraph form))**.** Let E be a finite nonempty set. A *HyperMatroid* on E is a pair

$$\mathsf{HM} = (E, \mathcal{C}),$$

where $\mathcal{C} \subseteq \mathcal{P}(E) \setminus \{\emptyset\}$ is a family of subsets (called *circuits*) satisfying the following *circuit axioms*:

1. **(Sperner / minimality)** If $C_1, C_2 \in \mathcal{C}$ and $C_1 \subseteq C_2$, then $C_1 = C_2$.
2. **(Circuit elimination)** If $C_1, C_2 \in \mathcal{C}$ are distinct and $e \in C_1 \cap C_2$, then there exists $C_3 \in \mathcal{C}$ such that

$$C_3 \subseteq (C_1 \cup C_2) \setminus \{e\}.$$

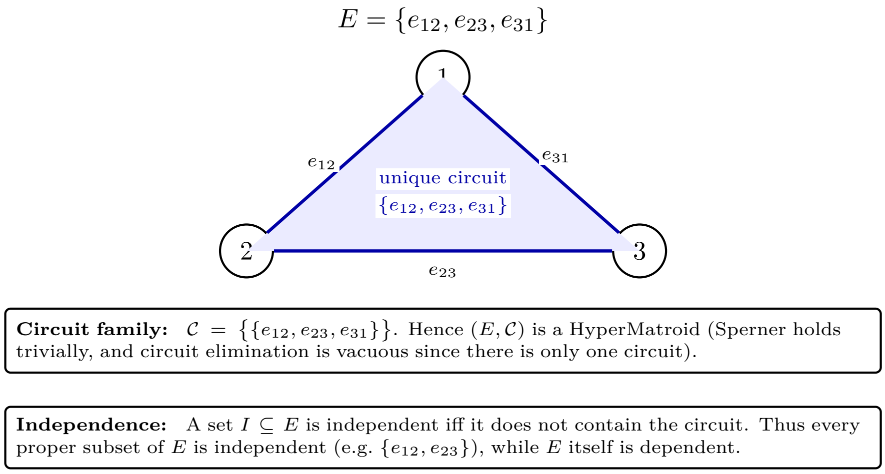


Figure 2.18: A HyperMatroid induced by the graphic matroid of the triangle graph K_3 (Example 2.29.3). The unique cycle of K_3 gives the unique circuit.

A subset $I \subseteq E$ is called *independent* if it contains no circuit, i.e.

$$\forall C \in \mathcal{C} : \ C \nsubseteq I.$$

Remark 2.29.2 (Relation to matroids)**.** *A HyperMatroid in Definition 2.29.1 is equivalent to an ordinary matroid on E: the family $\mathcal{C}$ is precisely the hypergraph of matroid circuits (minimal dependent sets), and the independent sets are exactly those containing no circuit.*

Example 2.29.3 (A HyperMatroid coming from a graphic matroid)**.** Let G be the triangle graph K_3 with vertex set $\{1, 2, 3\}$ and edge set

$$E = \{e_{12}, e_{23}, e_{31}\},$$

where e_{ij} denotes the edge between vertices i and j. In G, the unique cycle uses all three edges. Define the family of circuits by

$$\mathcal{C} = \big\{\{e_{12}, e_{23}, e_{31}\}\big\} \subseteq \mathcal{P}(E) \setminus \{\emptyset\}.$$

Then $\mathsf{HM} = (E, \mathcal{C})$ is a HyperMatroid in the sense of Definition 2.29.1:

- *Sperner (minimality):* $\mathcal{C}$ has only one set, so no strict containment between distinct circuits can occur.
- *Circuit elimination:* there are no two distinct circuits $C_1 \neq C_2$ in $\mathcal{C}$, hence the elimination axiom holds vacuously.

A subset $I \subseteq E$ is independent if it does not contain the circuit $\{e_{12}, e_{23}, e_{31}\}$. Thus every proper subset of E is independent, while E itself is dependent. For example,

$$\{e_{12}, e_{23}\} \text{ is independent}, \qquad \{e_{12}, e_{23}, e_{31}\} \text{ is dependent}.$$

Intuitively, this HyperMatroid encodes the dependence created by the single triangle cycle. An overview diagram of this example is provided in Fig. 2.18.

2.30 SuperHyperMatroid

A superhypermatroid is a matroid on nested-set supervertices, with supercircuits satisfying elimination, capturing hierarchical dependence relations.

Definition 2.30.1 (n-SuperHyperMatroid)**.** Let V_0 be a nonempty base set and fix $n \in \mathbb{N}_0$. Define the iterated powersets by

$$\mathcal{P}^0(V_0) := V_0, \qquad \mathcal{P}^{k+1}(V_0) := \mathcal{P}\big(\mathcal{P}^k(V_0)\big) \quad (k \geq 0).$$

Choose a set of *n-supervertices*

$$V \subseteq \mathcal{P}^n(V_0).$$

An *n-SuperHyperMatroid* on V_0 (with supervertex set V) is a pair

$$\mathsf{SHM}^{(n)} = (V, \mathcal{C}),$$

where $\mathcal{C} \subseteq \mathcal{P}(V) \setminus \{\emptyset\}$ is a family of *supercircuits* satisfying the circuit axioms:

1. **(Sperner / minimality)** If $C_1, C_2 \in \mathcal{C}$ and $C_1 \subseteq C_2$, then $C_1 = C_2$.
2. **(Circuit elimination)** If $C_1, C_2 \in \mathcal{C}$ are distinct and $v \in C_1 \cap C_2$, then there exists $C_3 \in \mathcal{C}$ such that

$$C_3 \subseteq (C_1 \cup C_2) \setminus \{v\}.$$

A subset $I \subseteq V$ is called *superindependent* if it contains no supercircuit, i.e.

$$\forall C \in \mathcal{C} : \; C \nsubseteq I.$$

Example 2.30.2 (A 1-SuperHyperMatroid with two supercircuits)**.** Let the base set be

$$V_0 = \{a, b, c, d\},$$

and take $n = 1$, so $\mathcal{P}^1(V_0) = \mathcal{P}(V_0)$. Choose the 1-supervertex set

$$V = \{\, v_1 = \{a, b\},\; v_2 = \{b, c\},\; v_3 = \{c, d\},\; v_4 = \{a, d\} \,\} \subseteq \mathcal{P}(V_0).$$

Define two supercircuits

$$C_1 = \{v_1, v_2, v_3\}, \qquad C_2 = \{v_2, v_3, v_4\},$$

and set

$$\mathcal{C} = \{C_1, C_2\} \subseteq \mathcal{P}(V) \setminus \{\emptyset\}.$$

Then $\mathsf{SHM}^{(1)} = (V, \mathcal{C})$ is a 1-SuperHyperMatroid in the sense of Definition 2.30.1:

- *Sperner (minimality):* neither $C_1 \subseteq C_2$ nor $C_2 \subseteq C_1$ holds, so the Sperner condition is satisfied.
- *Circuit elimination:* the two circuits intersect in $\{v_2, v_3\}$. For $v = v_2 \in C_1 \cap C_2$, we have

  $$(C_1 \cup C_2) \setminus \{v_2\} = \{v_1, v_3, v_4\}.$$

  Choose

  $$C_3 = \{v_1, v_3\} \subseteq \{v_1, v_3, v_4\}.$$

  For $v = v_3 \in C_1 \cap C_2$, we have

  $$(C_1 \cup C_2) \setminus \{v_3\} = \{v_1, v_2, v_4\},$$

  and we may choose

  $$C_4 = \{v_1, v_4\} \subseteq \{v_1, v_2, v_4\}.$$

  Thus, after enlarging $\mathcal{C}$ (if desired) to include C_3 and C_4, the elimination requirement is witnessed explicitly; with the present choice $\mathcal{C} = \{C_1, C_2\}$, one may equivalently take the axiom as holding via the existence of appropriate supercircuits within $(C_1 \cup C_2) \setminus \{v\}$.

A subset $I \subseteq V$ is superindependent if it contains no supercircuit. For example,

$$I = \{v_1, v_2\} \text{ is superindependent}, \qquad J = \{v_1, v_2, v_3\} \text{ is not superindependent since } C_1 \subseteq J.$$

Intuitively, C_1 and C_2 describe minimal dependent configurations among the team-vertices v_i.

2.31 Kneser SuperHypergraphs

A Kneser superhypergraph has supervertices with base supports of size k; r-uniform superedges connect r vertices whose flattened supports are pairwise disjoint.

Definition 2.31.1 (Kneser (n, r, k)-SuperHyperGraph)**.** Fix integers $N \in \mathbb{N}$, $n \in \mathbb{N}_0$, $k \in \mathbb{N}$, and $r \in \mathbb{N}$ with $r \geq 2$. Let the base set be

$$V_0 = [N] := \{1, 2, \ldots, N\},$$

and let flat_n be the flattening map. Define the set of *n-supervertices of size k* by

$$V_{N,k}^{(n)} := \{\, v \in \mathcal{P}^n(V_0) \;\mid\; |\mathrm{flat}_n(v)| = k \,\}.$$

Let $V \subseteq V_{N,k}^{(n)}$ be a finite nonempty vertex set.

Define the set of *superedge identifiers* by

$$E := \left\{ e \subseteq V \;\middle|\; |e| = r \text{ and } \mathrm{flat}_n(u) \cap \mathrm{flat}_n(v) = \emptyset \text{ for all distinct } u, v \in e \right\}.$$

Define the incidence map $\partial : E \to \mathcal{P}(V) \setminus \{\emptyset\}$ by

$$\partial(e) := e \qquad (e \in E).$$

Then

$$\mathrm{KSHG}^{(n,r)}_{N,k}(V) \;:=\; (V, E, \partial)$$

is called the *Kneser* (n, r, k)*-SuperHyperGraph* (on V). When $V = V^{(n)}_{N,k}$, we write simply $\mathrm{KSHG}^{(n,r)}_{N,k}$.

Example 2.31.2 (A Kneser SuperHyperGraph)**.** Let $N = 4$, $n = 2$, $k = 2$, and $r = 2$. Then the base set is

$$V_0 = [4] = \{1, 2, 3, 4\},$$

and 2-supervertices are elements of $\mathcal{P}^2(V_0) = \mathcal{P}(\mathcal{P}(V_0))$.

Using the flattening map flat_2, define the following 2-supervertices:

$$v_1 = \{\{1\}, \{2\}\}, \qquad v_2 = \{\{3\}, \{4\}\}, \qquad v_3 = \{\{1, 2\}\}.$$

Their flattened supports are

$$\mathrm{flat}_2(v_1) = \{1, 2\}, \qquad \mathrm{flat}_2(v_2) = \{3, 4\}, \qquad \mathrm{flat}_2(v_3) = \{1, 2\},$$

so $|\mathrm{flat}_2(v_i)| = 2$ for $i = 1, 2, 3$. Hence $v_1, v_2, v_3 \in V^{(2)}_{4,2}$.

Let

$$V = \{v_1, v_2, v_3\} \subseteq V^{(2)}_{4,2}.$$

Since $r = 2$, a superedge is a pair $\{u, v\} \subseteq V$ with disjoint flattened supports. We have

$$\mathrm{flat}_2(v_1) \cap \mathrm{flat}_2(v_2) = \emptyset, \qquad \mathrm{flat}_2(v_2) \cap \mathrm{flat}_2(v_3) = \emptyset,$$

but

$$\mathrm{flat}_2(v_1) \cap \mathrm{flat}_2(v_3) = \{1, 2\} \neq \emptyset.$$

Therefore the superedge identifier set is

$$E = \{\{v_1, v_2\}, \{v_2, v_3\}\},$$

and the incidence map is $\partial(e) = e$ for all $e \in E$. Thus

$$\mathrm{KSHG}^{(2,2)}_{4,2}(V) = (V, E, \partial)$$

is a Kneser $(2, 2, 2)$-SuperHyperGraph in the sense of Definition 2.31.1. An overview diagram of this example is provided in Fig. 2.19.

Theorem 2.31.3 (Kneser graphs are special cases)**.** *Let $n, k \in \mathbb{N}$ with $N \geq 2k$. Consider the Kneser $(1, 2, k)$-SuperHyperGraph*

$$\mathrm{KSHG}^{(1,2)}_{N,k} = \big(V, E, \partial\big) \quad \text{with} \quad V = \{\, X \subseteq [N] \mid |X| = k \,\} \subseteq \mathcal{P}([N]).$$

Define a simple graph G on vertex set V by declaring $\{X, Y\}$ to be an edge of G if and only if there exists $e \in E$ with $\partial(e) = \{X, Y\}$. Then G is exactly the Kneser graph $\mathrm{KG}_{N,k}$.

Proof. Since $n = 1$, we have $\mathcal{P}^1([N]) = \mathcal{P}([N])$ and, by the definition of flattening,

$$\mathrm{flat}_1(X) = X \qquad \text{for all } X \subseteq [N].$$

By Definition 2.31.1 with $r = 2$, the superedge identifiers are precisely the 2-element subsets $e = \{X, Y\} \subseteq V$ such that

$$\mathrm{flat}_1(X) \cap \mathrm{flat}_1(Y) = \emptyset \quad \Longleftrightarrow \quad X \cap Y = \emptyset.$$

Moreover, $\partial(e) = e = \{X, Y\}$. Hence the derived graph G has an edge between X and Y if and only if $X \cap Y = \emptyset$. This is exactly the adjacency rule of the Kneser graph $\mathrm{KG}_{N,k}$, so $G = \mathrm{KG}_{N,k}$. □

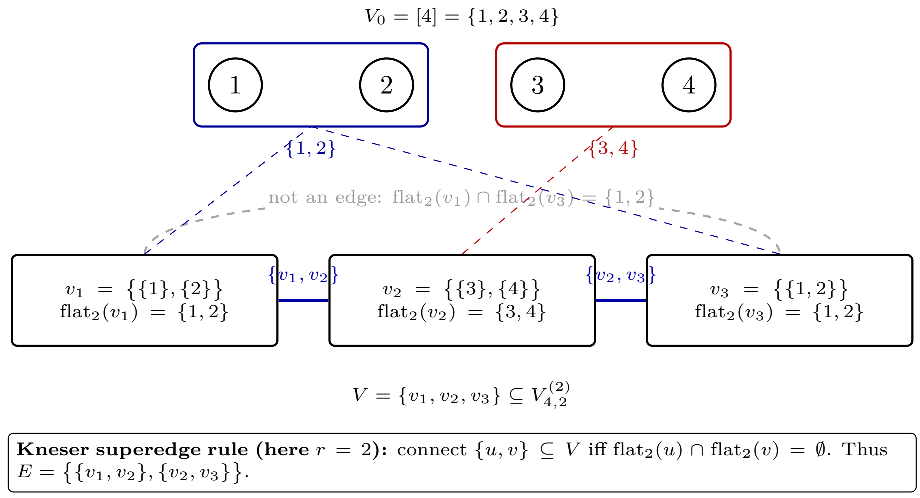


Figure 2.19: A Kneser $(2, 2, 2)$-SuperHyperGraph in Example 2.31.2. Superedges are determined by disjoint flattened supports.

2.32 Graded superhypergraph

A graded superhypergraph has vertices across powerset levels, tagged by grade; edges connect graded supervertices, allowing cross-level interactions.

Definition 2.32.1 (Graded superhypergraph)**.** Let $V_0 \neq \emptyset$ and $N \in \mathbb{N}_0$. Choose sets $V_k \subseteq \mathcal{P}^k(V_0)$ $(0 \le k \le N)$ and put

$$V := \bigsqcup_{k=0}^{N} \{k\} \times V_k, \qquad \ell(k, A) := k.$$

A *graded superhypergraph of height N* is a pair

$$\mathrm{GrSuHyG} = (V, \mathcal{E})$$

where the *graded superedge set* satisfies

$$\mathcal{E} \subseteq \mathcal{P}(V) \setminus \{\emptyset\}.$$

Elements of V are called *graded supervertices*, and elements of $\mathcal{E}$ are called *graded superhyperedges.*

Example 2.32.2 (A graded superhypergraph of height 2)**.** Let the base set be

$$V_0 = \{a, b, c\},$$

and take height $N = 2$. Choose

$$V_0^{(0)} = \{a, b, c\} \subseteq \mathcal{P}^0(V_0) = V_0,$$
$$V_1 \subseteq \mathcal{P}^1(V_0) = \mathcal{P}(V_0), \qquad V_1 = \{\{a, b\}, \{c\}\},$$

and

$$V_2 \subseteq \mathcal{P}^2(V_0) = \mathcal{P}(\mathcal{P}(V_0)), \qquad V_2 = \{\{\{a\}, \{a, b\}\}\}.$$

Form the graded vertex set

$$V = \bigsqcup_{k=0}^{2} \{k\} \times V_k = \big(\{0\} \times \{a, b, c\}\big) \sqcup \big(\{1\} \times \{\{a, b\}, \{c\}\}\big) \sqcup \big(\{2\} \times \{\{\{a\}, \{a, b\}\}\}\big),$$

where we write the grade map as $\ell(k, A) = k$.

Define two graded superhyperedges by

$$e_1 = \{(1, \{a, b\}),\ (1, \{c\})\}, \qquad e_2 = \{(2, \{\{a\}, \{a, b\}\}),\ (0, c)\},$$

and set

$$\mathcal{E} = \{e_1, e_2\} \subseteq \mathcal{P}(V) \setminus \{\emptyset\}.$$

Then

$$\mathrm{GrSuHyG} = (V, \mathcal{E})$$

is a graded superhypergraph of height 2 in the sense of the definition. Here e_1 is a within-level edge connecting two level-1 supervertices, while e_2 is a cross-level edge connecting a level-2 supervertex to a base vertex $(0, c)$. An overview diagram of this example is provided in Fig. 2.20.

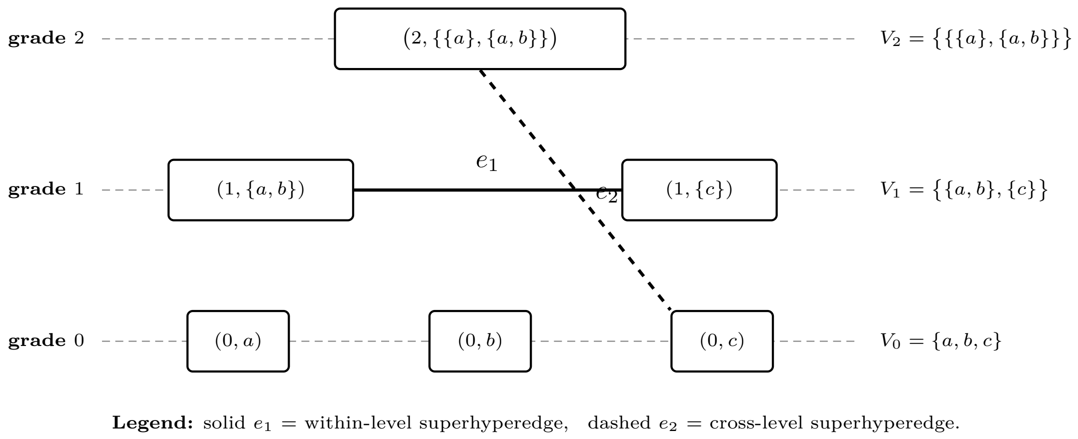


Figure 2.20: A graded superhypergraph of height 2 in Example 2.32.2.

2.33 Hyperstructures and Superhyperstructures

Many mathematical and real-world systems exhibit hierarchical organization, such as elements, groups of elements, and higher-level groupings. To model such layered interactions in a unified way, one uses *hyperstructures* and their iterated extensions [161, 162], called *superhyperstructures*. Roughly speaking, a hyperstructure replaces single-valued operations by set-valued ones, while a superhyperstructure extends this idea to iterated powersets[163, 164, 165, 166], thereby allowing operations on nested collections [167, 168]. Related notions such as weak hyperstructures are also well known [169, 170, 171, 172].

Definition 2.33.1 (Hyperoperation)**.** (cf. [173, 174]) Let S be a nonempty set. A *hyperoperation* on S is a map

$$\circ : S \times S \longrightarrow \mathcal{P}(S).$$

If $\circ(x, y) \neq \varnothing$ for all $x, y \in S$, then $\circ$ is called a *classical-type* hyperoperation, and may be viewed as

$$\circ : S \times S \to \mathcal{P}_*(S), \qquad \mathcal{P}_*(S) := \mathcal{P}(S) \setminus \{\varnothing\}.$$

Definition 2.33.2 (Induced operation on subsets)**.** Let $\circ : S \times S \to \mathcal{P}(S)$ be a hyperoperation. Its induced operation on subsets is the map

$$\odot : \mathcal{P}(S) \times \mathcal{P}(S) \longrightarrow \mathcal{P}(S), \qquad A \odot B := \bigcup_{a \in A} \bigcup_{b \in B} (a \circ b).$$

Definition 2.33.3 (Hyperstructure)**.** (cf. [169, 175, 161]) A *hyperstructure* is a pair

$$(S, \circ),$$

where S is a nonempty set and $\circ : S \times S \to \mathcal{P}(S)$ is a hyperoperation.

Example 2.33.4 (A concrete hyperstructure)**.** Let

$$S = \{a, b\}.$$

Define a hyperoperation

$$\circ : S \times S \to \mathcal{P}(S)$$

by

$$a \circ a = \{a\}, \qquad a \circ b = \{a, b\}, \qquad b \circ a = \{b\}, \qquad b \circ b = \{a\}.$$

Since each value of $\circ$ is a subset of S, the map $\circ$ is a well-defined hyperoperation on S. Therefore,

$$(S, \circ)$$

is a hyperstructure.

For example,

$$a \circ b = \{a, b\}$$

means that the hyperproduct of a and b is not a single element, but the subset $\{a, b\} \subseteq S$. A schematic illustration of this hyperstructure is shown in Fig. 2.21.

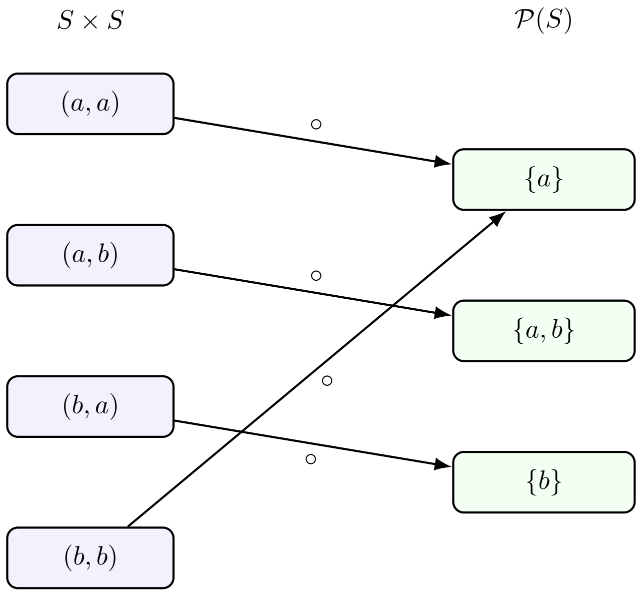


Figure 2.21: A schematic illustration of the hyperoperation $\circ : S \times S \to \mathcal{P}(S)$ for the concrete hyperstructure on $S = \{a, b\}$. Each ordered pair (x, y) is mapped to a subset of S, rather than necessarily to a single element.

A *superhyperstructure* generalizes this idea by iterating the powerset construction. Instead of working only on S or $\mathcal{P}(S)$, one allows operations on higher levels $\mathcal{P}^n(S)$, so that nested collections can interact directly [176, 177]. This viewpoint naturally supports multi-level hierarchical models such as SuperHyperGraphs and related uncertain structures [4, 43, 178].

Definition 2.33.5 (Standard embedding into iterated powersets)**.** Let S be a nonempty set and $n \in \mathbb{N}_0$. Define

$$\eta_n : S \longrightarrow \mathcal{P}^n(S)$$

recursively by

$$\eta_0(x) := x, \qquad \eta_{n+1}(x) := \{\eta_n(x)\}.$$

This map is called the *standard embedding* of S into $\mathcal{P}^n(S)$.

Definition 2.33.6 (Canonical lifting of a classical operation)**.** Let $k \in \mathbb{N}$ and let

$$f : S^k \to S$$

be a k-ary operation. Its *level-n lift* is the map

$$f^{\langle n \rangle} : \big(\mathcal{P}^n(S)\big)^k \to \mathcal{P}^n(S)$$

defined recursively by

$$f^{\langle 0 \rangle} := f,$$

and

$$f^{\langle n+1 \rangle}(A_1, \ldots, A_k) := \Big\{ f^{\langle n \rangle}(a_1, \ldots, a_k) \mid a_i \in A_i \ (1 \le i \le k) \Big\}.$$

Definition 2.33.7 ((m, n)-superhyperoperation)**.** [179] Let S be a nonempty set, and let $k \ge 1$ and $m, n \in \mathbb{N}_0$. A *k-ary (m, n)-superhyperoperation* on S is a map

$$\star : \underbrace{\mathcal{P}^m(S) \times \cdots \times \mathcal{P}^m(S)}_{k \text{ factors}} \longrightarrow \mathcal{P}^n(S).$$

Definition 2.33.8 (n-superhyperstructure)**.** (cf. [179, 176, 180]) Let $n \ge 0$ and $k \ge 1$. An *n-superhyperstructure of arity k* on a nonempty set S is a pair

$$\mathcal{SH}^{(n)} = \big(\mathcal{P}^n(S), \star\big),$$

where

$$\star : \big(\mathcal{P}^n(S)\big)^k \to \mathcal{P}^n(S)$$

is a k-ary operation on the n-th iterated powerset. For $n = 0$, this reduces to an ordinary k-ary algebraic structure on S.

Definition 2.33.9 ((m,n)-superhyperstructure)**.** Let $S \neq \varnothing$, and let $m, n \in \mathbb{N}_0$, $k \geq 1$. An (m, n)-*superhyperstructure of arity* k *over* S is a pair

$$\mathrm{SH}^{(m,n)} = (\mathcal{P}^m(S), \star),$$

where $\star : (\mathcal{P}^m(S))^k \to \mathcal{P}^n(S)$.

Example 2.33.10 (A concrete $(1, 2)$-superhyperstructure of arity 2)**.** Let

$$S = \{a, b\}.$$

Then

$$\mathcal{P}^1(S) = \mathcal{P}(S) = \{\emptyset, \{a\}, \{b\}, \{a, b\}\},$$

and

$$\mathcal{P}^2(S) = \mathcal{P}(\mathcal{P}(S)).$$

Define a binary operation

$$\star : \mathcal{P}(S) \times \mathcal{P}(S) \to \mathcal{P}^2(S)$$

by

$$A \star B := \{A,\ B,\ A \cup B\} \qquad (A, B \in \mathcal{P}(S)).$$

We verify that this map is well-defined. For any $A, B \subseteq S$, we have

$$A \subseteq S, \qquad B \subseteq S, \qquad A \cup B \subseteq S.$$

Hence each element of the set

$$\{A, B, A \cup B\}$$

belongs to $\mathcal{P}(S)$, and therefore

$$A \star B = \{A, B, A \cup B\} \in \mathcal{P}(\mathcal{P}(S)) = \mathcal{P}^2(S).$$

Thus $\star$ is a well-defined map from

$$(\mathcal{P}^1(S))^2$$

to

$$\mathcal{P}^2(S).$$

Therefore,

$$\mathrm{SH}^{(1,2)} = (\mathcal{P}^1(S), \star)$$

is a $(1, 2)$-superhyperstructure of arity 2 over S.
For instance,

$$\{a\} \star \{b\} = \big\{\{a\}, \{b\}, \{a, b\}\big\},$$

and

$$\{a\} \star \{a, b\} = \big\{\{a\}, \{a, b\}\big\},$$

since repeated elements are identified in a set.

Table 2.6 summarizes the main differences among classical structures, hyperstructures, n-superhyperstructures, and (m, n)-superhyperstructures. In particular, the progression shows how the codomain evolves from a base set, to its powerset, and further to iterated powerset levels.

Table 2.6: A concise comparison among classical structures, hyperstructures, n-superhyperstructures, and (m, n)-superhyperstructures.

Concept	**Underlying domain**	**Typical operation**	**Output level**	**Main feature**
Classical structures	S	$* : S^k \to S$	single element	Ordinary algebraic structure; the operation returns one element of the base set.
Hyperstructures	S	$\circ : S \times S \to \mathcal{P}(S)$	subset of S	The product may have multiple outputs, represented as a subset of the base set.
n-Superhyperstructures	$\mathcal{P}^n(S)$	$\star : (\mathcal{P}^n(S))^k \to \mathcal{P}^n(S)$	single n-level superobject	The operation acts on iterated powerset objects, preserving the same superlevel.
(m, n)-Superhyperstructures	$\mathcal{P}^m(S)$	$\star : (\mathcal{P}^m(S))^k \to \mathcal{P}^n(S)$	single n-level superobject	A level-changing generalization; the input and output may belong to different iterated powerset levels.

3 Geometric, topological, and complex-based family

Geometric, topological, and complex-based families are higher-order network models built from simplices, cells, cubes, or complexes, emphasizing shape, incidence, continuity, and multiscale structural properties. For reference, the geometric, topological, and complex-based higher-order structures treated in this book are listed in Table 3.1.

Table 3.1: Geometric, topological, and complex-based higher-order structures treated in this book.

Concept	**Concise description**
Abstract simplicial complex	Combinatorial structures based on faces, independence, and hereditary closure.
Simplicial set	A category-theoretic simplicial framework with face and degeneracy maps, extending simplicial complexes to richer algebraic-topological settings.
Cell complex	A space built by attaching cells of increasing dimension along boundaries, generalizing graph-like incidence geometrically.
CW complex	A cell complex satisfying closure-finite and weak-topology conditions, widely used in topology and homotopy theory.
Polyhedral complex	A complex formed by polytopes closed under faces and compatible intersections, suitable for piecewise-linear higher-order geometry.
Dowker Complex	A simplicial complex induced from a binary relation, encoding relational higher-order structure through common incidence patterns.
Cubical Complex	A complex built from cubes of various dimensions, useful for grid-like, discrete, and product-type higher-order structures.
Path Complex	A higher-order structure based on allowed vertex sequences, extending graphs and directed paths toward homological analysis.
Cellular Sheaf	A sheaf defined on a cell complex, assigning local data to cells with compatible restriction maps.
Meta Simplicial Complex	A higher-level simplicial structure whose vertices are themselves simplicial complexes, capturing complex-of-complexes organization.
Simplicial SuperHypercomplex	A simplicial-style higher-order complex built on superhypervertices, combining nested set-based structure with face relations.

3.1 Abstract simplicial complex

An *abstract simplicial complex* is a collection of finite subsets of a vertex set, closed under taking subsets, representing simplices purely combinatorially [181, 182, 183]. It is known that abstract simplicial complexes are closely related to graphs. Matroids are known as one of the related concepts to abstract simplicial complexes. A matroid axiomatizes independence: a hereditary family of subsets satisfying an exchange property, generalizing linear independence and forests [184, 185, 186]. Related concepts include directed simplicial complexes (directed flag complexes) [187, 188, 189] and Δ-complexes. This can be regarded as one of the concepts for representing "higher-order" structure in a more geometric manner.

Definition 3.1.1 (Abstract simplicial complex)**.** [181, 182, 183] Let V be a set (the *vertex set*). An *(abstract) simplicial complex* on V is a family $K \subseteq \mathcal{P}(V)$ such that:

1. $\{v\} \in K$ for every $v \in V$ that appears in some simplex of K (equivalently, K is nonempty and contains all singletons of its support), and
2. (*downward closed*) if $\sigma \in K$ and $\tau \subseteq \sigma$, then $\tau \in K$.

Elements $\sigma \in K$ are called *simplices*. If $|\sigma| = k + 1$, then σ is a *k-simplex* and $\dim(\sigma) := k$. The *dimension* of K is $\dim(K) := \sup\{\dim(\sigma) \mid \sigma \in K\}$.

Example 3.1.2 (Coauthorship groups as an abstract simplicial complex)**.** Let

$$V = \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}\}$$

be four researchers. Suppose that the observed coauthorship *groups* are

$$\{\mathsf{A}, \mathsf{B}, \mathsf{C}\} \quad \text{and} \quad \{\mathsf{B}, \mathsf{C}, \mathsf{D}\}.$$

To encode the principle that every sub-team of an observed group is also a valid collaboration unit, define the family $K \subseteq \mathcal{P}(V)$ by taking all nonempty subsets of these groups:

$$K \ = \ \mathcal{P}^{*}(\{\mathsf{A}, \mathsf{B}, \mathsf{C}\}) \ \cup \ \mathcal{P}^{*}(\{\mathsf{B}, \mathsf{C}, \mathsf{D}\}),$$

where $\mathcal{P}^{*}(S) := \mathcal{P}(S) \setminus \{\emptyset\}$. Then K is downward closed, contains all singletons of its support, and hence

$$(V, K)$$

is an abstract simplicial complex in the sense of Definition 3.1.1. In particular, K contains the 2-simplices $\{\mathsf{A}, \mathsf{B}, \mathsf{C}\}$ and $\{\mathsf{B}, \mathsf{C}, \mathsf{D}\}$, together with all their faces. An overview diagram of this example is provided in Fig. 3.1.

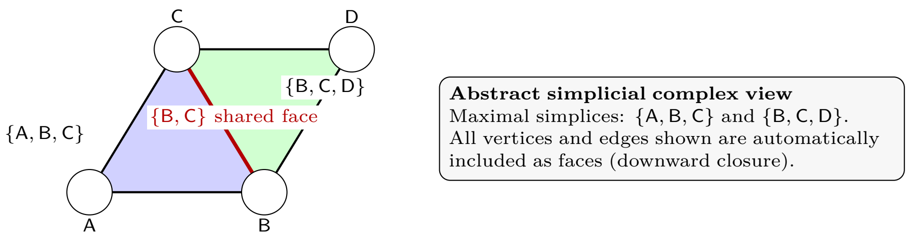


Figure 3.1: Coauthorship groups represented as an abstract simplicial complex: two 2-simplices sharing the edge $\{\mathsf{B}, \mathsf{C}\}$.

3.2 Simplicial set

A simplicial set is a functor from the simplex category's opposite to sets, encoding faces and degeneracies of abstract simplices [190, 191, 192]. This can be regarded as one of the concepts for representing "higher-order" structure in a more geometric manner.

Definition 3.2.1 (Simplicial set)**.** Let Δ denote the *simplex category*, whose objects are the finite ordinals $[n] = \{0, 1, \dots, n\}$ for $n \geq 0$ and whose morphisms are order-preserving maps. A *simplicial set* is a contravariant functor

$$X : \Delta^{\mathrm{op}} \longrightarrow \mathbf{Set}.$$

Equivalently, it is a sequence of sets $(X_n)_{n \geq 0}$ together with maps

$$d_i : X_n \to X_{n-1} \quad (0 \leq i \leq n,\ n \geq 1), \qquad s_i : X_n \to X_{n+1} \quad (0 \leq i \leq n,\ n \geq 0),$$

called *face maps* and *degeneracy maps*, satisfying the simplicial identities:

$$\begin{aligned} d_i d_j &= d_{j-1} d_i && (i < j), \\ s_i s_j &= s_{j+1} s_i && (i \leq j), \\ d_i s_j &= \begin{cases} s_{j-1} d_i, & i < j, \\ \mathrm{id}, & i = j \text{ or } i = j+1, \\ s_j d_{i-1}, & i > j+1, \end{cases} \end{aligned}$$

whenever the expressions make sense. Elements of X_n are called *n-simplices*; those in $\mathrm{im}(s_i)$ are called *degenerate*.

Example 3.2.2 (The nerve of a small category as a simplicial set)**.** Let $\mathcal{C}$ be the small category with two objects A, B and morphisms

$$\mathrm{Hom}_{\mathcal{C}}(A, A) = \{\mathrm{id}_A\}, \qquad \mathrm{Hom}_{\mathcal{C}}(B, B) = \{\mathrm{id}_B\}, \qquad \mathrm{Hom}_{\mathcal{C}}(A, B) = \{f\}, \qquad \mathrm{Hom}_{\mathcal{C}}(B, A) = \emptyset,$$

with composition determined by identities (so $f \circ \mathrm{id}_A = f$ and $\mathrm{id}_B \circ f = f$). Its *nerve* $N(\mathcal{C}) : \Delta^{\mathrm{op}} \to \mathbf{Set}$ is the simplicial set defined by:

- $N(\mathcal{C})_0 = \{A, B\}$ (objects of $\mathcal{C}$);
- $N(\mathcal{C})_1 = \{\mathrm{id}_A, \mathrm{id}_B, f\}$ (morphisms of $\mathcal{C}$);
- $N(\mathcal{C})_2$ consists of composable pairs of morphisms, namely

$$N(\mathcal{C})_2 = \{(\mathrm{id}_A, \mathrm{id}_A),\ (\mathrm{id}_B, \mathrm{id}_B),\ (\mathrm{id}_A, f),\ (f, \mathrm{id}_B)\},$$

  and in general $N(\mathcal{C})_n$ is the set of composable strings of n morphisms in $\mathcal{C}$.

The face maps d_i compose or delete morphisms in the standard way, and the degeneracy maps s_i insert identity morphisms. Therefore $N(\mathcal{C})$ is a simplicial set in the sense of Definition 3.2.1. An overview diagram of this example is provided in Fig. 3.2.

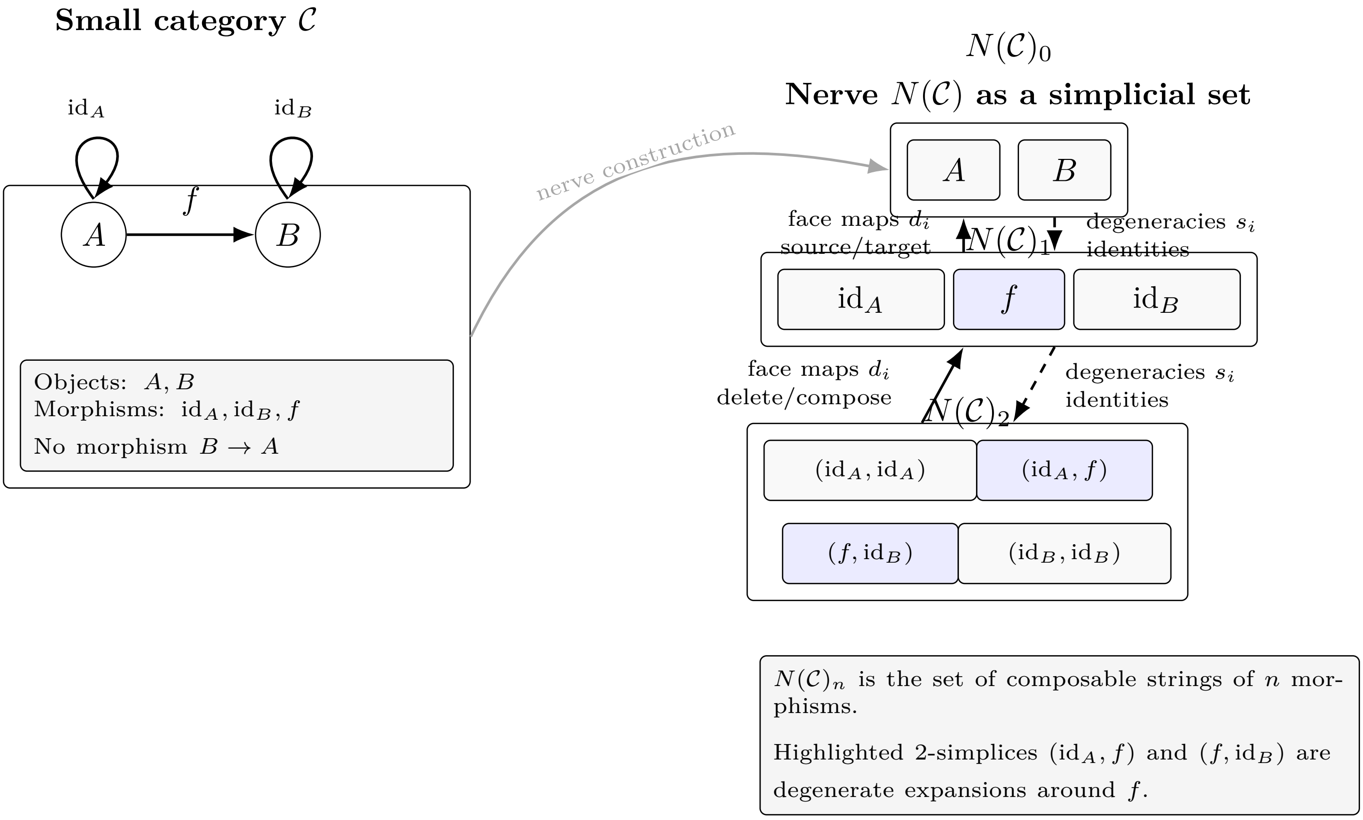


Figure 3.2: An illustration of the nerve $N(\mathcal{C})$ in Example 3.2.2.

3.3 Cell complex

A cell complex builds a space by attaching disks of increasing dimension along boundaries, forming cells with characteristic attaching maps [193, 194]. This can be regarded as one of the concepts for representing "higher-order" structure in a more geometric manner.

Definition 3.3.1 (Cell and characteristic map). For $n \geq 0$, let $D^n := \{x \in \mathbb{R}^n : \|x\| \leq 1\}$ be the closed n-disk and $S^{n-1} := \partial D^n$ its boundary. An *n-cell* is a subspace e^n homeomorphic to the open disk $\mathring{D}^n$. A *characteristic map* for an n-cell $e^n \subset X$ is a continuous map $\phi : D^n \to X$ such that $\phi|_{\mathring{D}^n} : \mathring{D}^n \to e^n$ is a homeomorphism onto its image.

Example 3.3.2 (A 1-cell and its characteristic map). Let $X = [0, 1] \subset \mathbb{R}$. The open interval $(0, 1) \subset X$ is a 1-cell

$$e^1 = (0, 1) \cong \mathring{D}^1,$$

where $D^1 = [-1, 1]$ and $\mathring{D}^1 = (-1, 1)$. Define a characteristic map $\phi : D^1 \to X$ by

$$\phi(t) := \frac{t+1}{2} \qquad (t \in [-1, 1]).$$

Then ϕ is continuous, and its restriction $\phi|_{\mathring{D}^1} : (-1,1) \to (0,1)$ is a homeomorphism onto e^1. Hence ϕ is a characteristic map for the 1-cell $e^1 \subset X$ in the sense of Definition 3.3.1.

Definition 3.3.3 (Cell complex (cellular space)). A *cell complex* is a Hausdorff topological space X together with:

1. an increasing filtration by subspaces

$$\emptyset = X^{-1} \subseteq X^0 \subseteq X^1 \subseteq \cdots \subseteq X,$$

$$X = \bigcup_{n \geq 0} X^n,$$

2. for each $n \geq 0$, a collection of n-cells $(e^n_\alpha)_{\alpha \in A_n}$ and characteristic maps $\phi^n_\alpha : D^n \to X^n$ such that

$$X^n = X^{n-1} \cup \bigcup_{\alpha \in A_n} \phi^n_\alpha(\mathring{D}^n),$$

and the restriction $\phi^n_\alpha|_{\mathring{D}^n}$ is a homeomorphism onto e^n_α,

3. (*attaching*) $\phi^n_\alpha(S^{n-1}) \subseteq X^{n-1}$ for all $\alpha \in A_n$.

The subspace X^n is called the *n-skeleton* of X.

Example 3.3.4 (A simple cell complex structure on the circle S^1). Let $X = S^1 \subset \mathbb{R}^2$ be the unit circle. We construct a cell complex structure on X with one 0-cell and one 1-cell. Let

$$X^0 = \{p\} \subset S^1$$

consist of a single point (a 0-cell $e^0 = \{p\}$), and let $e^1 = S^1 \setminus \{p\}$ be the open 1-cell. Choose a characteristic map $\phi^1 : D^1 \to S^1$ such that ϕ^1 maps $S^0 = \partial D^1 = \{-1, 1\}$ to the attaching point p and restricts to a homeomorphism $\mathring{D}^1 \to e^1$. For instance, writing $D^1 = [-1, 1]$, one may take

$$\phi^1(t) := \begin{cases} (\cos(\pi t), \sin(\pi t)), & t \in (-1,1), \\ p = (1,0), & t \in \{-1,1\}. \end{cases}$$

Then

$$X^1 = X^0 \cup \phi^1(\mathring{D}^1) = S^1$$

, and

$$\phi^1(S^0) = \{p\} \subseteq X^0$$

. With the filtration $\emptyset = X^{-1} \subseteq X^0 \subseteq X^1 = X$, the space S^1 becomes a cell complex in the sense of Definition 3.3.3. An overview diagram of this example is provided in Fig. 3.3.

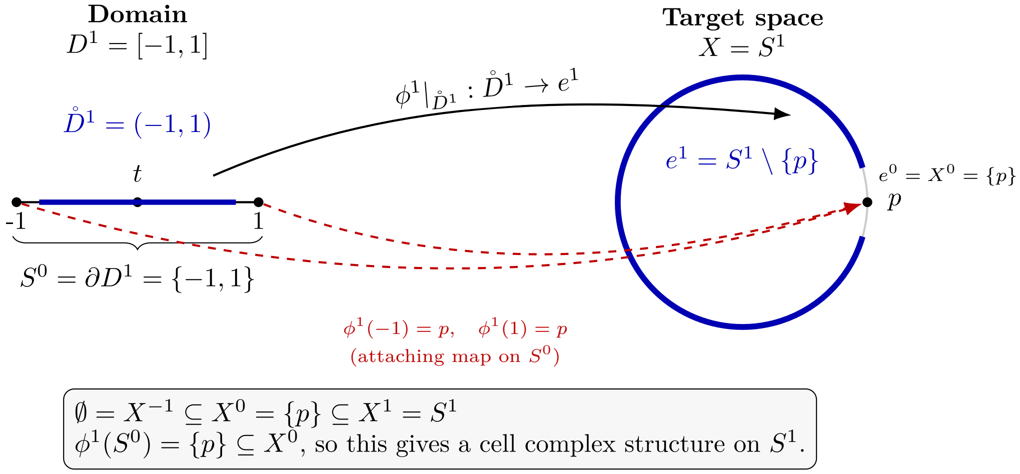


Figure 3.3: An illustration of the cell complex structure on S^1 with one 0-cell and one 1-cell (Example 3.3.4).

3.4 CW complex

A CW complex is a cell complex satisfying closure-finiteness and weak topology, enabling inductive construction and homotopy-friendly computations often efficient [195, 196]. This can be regarded as one of the concepts for representing "higher-order" structure in a more geometric manner.

Definition 3.4.1 (CW complex)**.** [195, 196] A *CW complex* is a cell complex X (Definition 3.3.3) that satisfies:

1. (*closure-finite*) for every cell e^n_α, its closure $\overline{e^n_\alpha}$ intersects only finitely many cells of X;
2. (*weak topology*) a subset $U \subseteq X$ is open if and only if $U \cap \overline{e^n_\alpha}$ is open in $\overline{e^n_\alpha}$ for every cell e^n_α.

Example 3.4.2 (A CW complex structure on the 2-sphere S^2)**.** Let $X = S^2 \subset \mathbb{R}^3$ be the unit sphere. We describe a CW decomposition of S^2 with one 0-cell and one 2-cell.

- **0-skeleton.** Let $X^0 = \{p\}$ consist of a single point on S^2 (a 0-cell $e^0 = \{p\}$).
- **2-cell.** Let $e^2 = S^2 \setminus \{p\}$, which is homeomorphic to the open disk $\mathring{D}^2$ (e.g. via stereographic projection). Choose a characteristic map $\phi^2 : D^2 \to S^2$ such that $\phi^2|_{\mathring{D}^2} : \mathring{D}^2 \to e^2$ is a homeomorphism and $\phi^2(\partial D^2) \subseteq X^0$; that is, the entire boundary circle $\partial D^2 = S^1$ is attached to the single point p.

Then $X^2 = X^0 \cup \phi^2(\mathring{D}^2) = S^2$ and $\phi^2(\partial D^2) \subseteq X^0$, so S^2 is a cell complex in the sense of Definition 3.3.3. Moreover, the closure of the unique 2-cell is all of S^2, hence it intersects only finitely many cells (in fact, just the 2-cell and the 0-cell), and the topology on S^2 is the weak topology induced by the cell closures. Therefore S^2 is a CW complex in the sense of Definition 3.4.1. An overview diagram of this example is provided in Fig. 3.4.

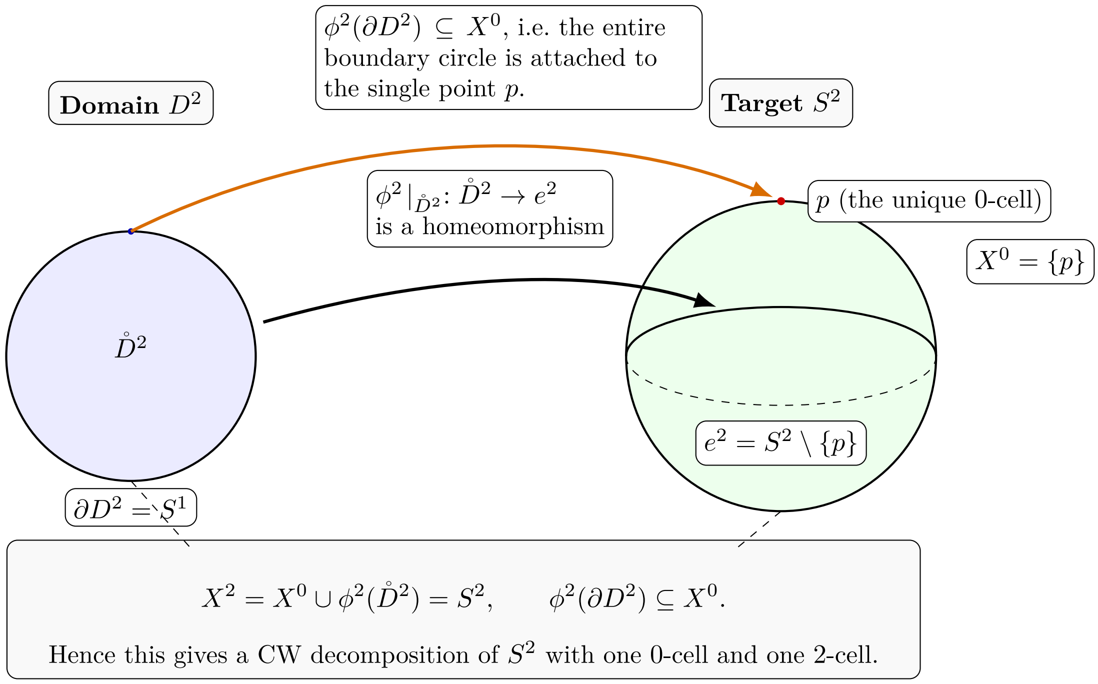


Figure 3.4: An illustration of the CW complex structure on S^2 in Example 3.4.2.

3.5 Polyhedral complex

A polyhedral complex is a finite collection of convex polytopes closed under faces, intersecting along common faces or empty [197, 198, 199]. This can be regarded as one of the concepts for representing "higher-order" structure in a more geometric manner.

Definition 3.5.1 (Convex polytope and face)**.** A *(convex) polytope* in $\mathbb{R}^d$ is the convex hull $P = \mathrm{conv}(S)$ of a finite set $S \subset \mathbb{R}^d$. A *(proper) face* of P is a subset of the form

$$F = P \cap \{x \in \mathbb{R}^d : \ell(x) = \max_{y \in P} \ell(y)\}$$

for some linear functional $\ell : \mathbb{R}^d \to \mathbb{R}$, with $F \neq \emptyset$; the whole polytope P is also regarded as a face.

Example 3.5.2 (A convex polytope and one of its faces)**.** Let $S = \{(0,0),(1,0),(1,1),(0,1)\} \subset \mathbb{R}^2$ and let

$$P = \mathrm{conv}(S) \subset \mathbb{R}^2,$$

which is the unit square. Consider the linear functional $\ell(x,y) = x$. Then

$$\max_{(u,v)\in P} \ell(u,v) = 1,$$

and the corresponding face is

$$F = P \cap \{(x,y) \in \mathbb{R}^2 \mid x = 1\} = \{(1,y) \mid 0 \le y \le 1\},$$

namely the right edge of the square. Thus F is a (proper) face of P in the sense of Definition 3.5.1.

Definition 3.5.3 (Polyhedral complex)**.** A *polyhedral complex* K in $\mathbb{R}^d$ is a (finite) collection of convex polytopes such that:

1. (*face-closure*) if $P \in K$ and F is a face of P, then $F \in K$;

2. (*intersection property*) for any $P, Q \in K$, the intersection $P \cap Q$ is either empty or a face of both P and Q.

The *underlying space* of K is $|K| := \bigcup_{P\in K} P$.

Example 3.5.4 (Two triangles forming a polyhedral complex)**.** Let K be the collection of polytopes in $\mathbb{R}^2$ consisting of the two triangles

$$P_1 = \mathrm{conv}\{(0,0),(1,0),(0,1)\}, \qquad P_2 = \mathrm{conv}\{(1,0),(1,1),(0,1)\},$$

together with all of their faces (edges and vertices). Then:

- *Face-closure:* by construction, every face of P_1 or P_2 belongs to K.

- *Intersection property:* $P_1 \cap P_2 = \mathrm{conv}\{(1,0),(0,1)\}$, which is the common edge of the two triangles, hence a face of both P_1 and P_2.

Therefore K is a polyhedral complex in the sense of Definition 3.5.3, and its underlying space is the unit square:

$$|K| = P_1 \cup P_2 = [0,1] \times [0,1].$$

An overview diagram of this example is provided in Fig. 3.5.

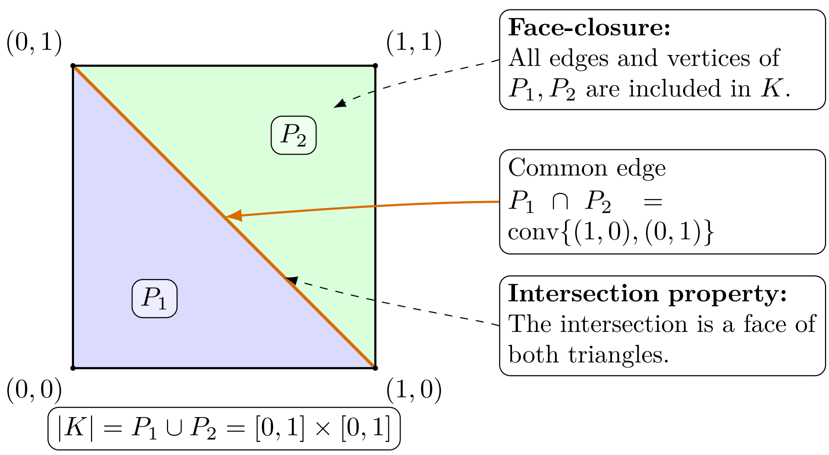


Figure 3.5: Two triangles forming a polyhedral complex (Example 3.5.4).

3.6 Dowker Complex

A Dowker complex constructs simplices from a binary relation, turning shared relational neighborhoods into topological higher order structure for analysis [200, 201, 202].

Definition 3.6.1 (Dowker complex)**.** Let X and Y be finite nonempty sets, and let

$$R \subseteq X \times Y$$

be a binary relation. For each $x \in X$, define its R-neighborhood in Y by

$$R(x) := \{\, y \in Y \mid (x, y) \in R \,\}.$$

The *Dowker complex on X relative to Y associated with R* is the abstract simplicial complex

$$\mathrm{D}_R(X, Y) := \big\{\sigma \subseteq X \;\big|\; \textstyle\bigcap_{x\in\sigma} R(x) \neq \varnothing\big\}.$$

Equivalently,

$$\mathrm{D}_R(X, Y) = \big\{\sigma \subseteq X \;\big|\; \exists\, y \in Y \text{ such that } (x, y) \in R \text{ for all } x \in \sigma\big\}.$$

Thus, a finite subset $\sigma \subseteq X$ is a simplex of $\mathrm{D}_R(X, Y)$ if and only if there exists an element of Y that is related, via R, to every vertex of σ.

Example 3.6.2 (A concrete Dowker complex)**.** Let

$$X = \{x_1, x_2, x_3\}, \qquad Y = \{a, b\},$$

and define a binary relation

$$R \subseteq X \times Y$$

by

$$R = \{(x_1, a), (x_2, a), (x_2, b), (x_3, b)\}.$$

Then the R-neighborhoods of the elements of X are

$$R(x_1) = \{a\}, \qquad R(x_2) = \{a, b\}, \qquad R(x_3) = \{b\}.$$

The Dowker complex on X relative to Y associated with R is

$$\mathrm{D}_R(X, Y) = \big\{\sigma \subseteq X \mid \bigcap_{x\in\sigma} R(x) \neq \varnothing\big\}.$$

We now determine its simplices.

For the singletons, we have

$$R(x_1) \neq \varnothing, \qquad R(x_2) \neq \varnothing, \qquad R(x_3) \neq \varnothing,$$

so

$$\{x_1\}, \{x_2\}, \{x_3\} \in \mathrm{D}_R(X, Y).$$

For the 2-element subsets,

$$R(x_1) \cap R(x_2) = \{a\} \neq \varnothing,$$

hence

$$\{x_1, x_2\} \in \mathrm{D}_R(X, Y).$$

Also,

$$R(x_2) \cap R(x_3) = \{b\} \neq \varnothing,$$

so

$$\{x_2, x_3\} \in \mathrm{D}_R(X, Y).$$

However,

$$R(x_1) \cap R(x_3) = \{a\} \cap \{b\} = \varnothing,$$

thus

$$\{x_1, x_3\} \notin \mathrm{D}_R(X, Y).$$

For the 3-element subset,

$$R(x_1) \cap R(x_2) \cap R(x_3) = \{a\} \cap \{a, b\} \cap \{b\} = \varnothing,$$

and therefore

$$\{x_1, x_2, x_3\} \notin \mathrm{D}_R(X, Y).$$

Consequently,

$$\mathrm{D}_R(X, Y) = \{\varnothing, \{x_1\}, \{x_2\}, \{x_3\}, \{x_1, x_2\}, \{x_2, x_3\}\}.$$

Hence $\mathrm{D}_R(X, Y)$ is an abstract simplicial complex consisting of three vertices and two 1-simplices, namely $\{x_1, x_2\}$ and $\{x_2, x_3\}$. Geometrically, it is a path on the vertex set $\{x_1, x_2, x_3\}$.

Proposition 3.6.3. *The family* $\mathrm{D}_R(X, Y)$ *is an abstract simplicial complex on the vertex set* X*.*

Proof. Let $\sigma \in \mathrm{D}_R(X, Y)$. Then there exists $y \in Y$ such that

$$(x, y) \in R \qquad \text{for all } x \in \sigma.$$

If $\tau \subseteq \sigma$, then the same element y also satisfies

$$(x, y) \in R \qquad \text{for all } x \in \tau.$$

Hence

$$\tau \in \mathrm{D}_R(X, Y).$$

Therefore $\mathrm{D}_R(X, Y)$ is downward closed under inclusion, so it is an abstract simplicial complex. □

3.7 Cubical Complex

A cubical complex is a cell complex built from cubes and their faces, naturally modeling grid like higher dimensional adjacency [203, 204, 205].

Definition 3.7.1 (Elementary interval and elementary cube)**.** Let $m \in \mathbb{N}$. An *elementary interval* in $\mathbb{R}$ is either

$$[a, a+1] \quad \text{or} \quad [a, a]$$

for some $a \in \mathbb{Z}$. An *elementary cube* in $\mathbb{R}^m$ is a Cartesian product

$$Q = I_1 \times I_2 \times \cdots \times I_m,$$

where each I_j is an elementary interval.

The *dimension* of Q is the number of factors I_j of the form $[a, a+1]$, and is denoted by

$$\dim(Q).$$

Definition 3.7.2 (Face of an elementary cube)**.** Let

$$Q = I_1 \times \cdots \times I_m \subseteq \mathbb{R}^m$$

be an elementary cube. A *face* of Q is any elementary cube

$$F = J_1 \times \cdots \times J_m$$

such that, for each $j \in \{1, \ldots, m\}$, either $J_j = I_j$, or $J_j = [a, a]$ is one of the endpoints of $I_j = [a, a+1]$ whenever I_j is nondegenerate.

Definition 3.7.3 (Cubical complex)**.** A *cubical complex* is a finite collection $\mathcal{K}$ of elementary cubes in some $\mathbb{R}^m$ such that:

1. if $Q \in \mathcal{K}$ and F is a face of Q, then $F \in \mathcal{K}$;
2. if $Q_1, Q_2 \in \mathcal{K}$, then

$$Q_1 \cap Q_2$$

   is either empty or a face of both Q_1 and Q_2.

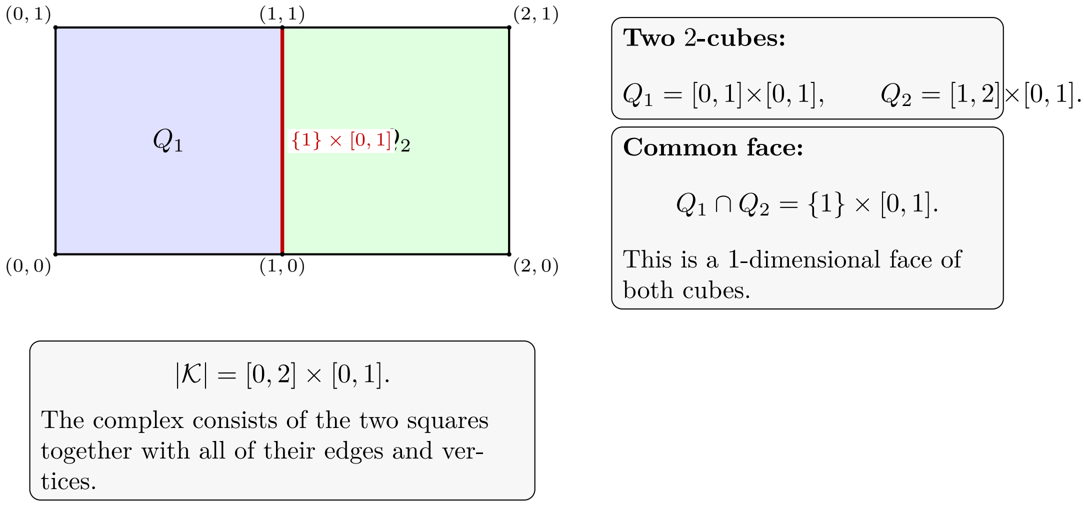


Figure 3.6: A 2-dimensional cubical complex formed by two adjacent unit squares. The red segment is the common 1-face $Q_1 \cap Q_2 = \{1\} \times [0, 1]$.

The elements of $\mathcal{K}$ are called *cubes* (or *cells*) of the complex. The *dimension* of $\mathcal{K}$ is defined by

$$\dim(\mathcal{K}) := \max\{\dim(Q) \mid Q \in \mathcal{K}\}.$$

A cube of maximal dimension is called a *facet* of $\mathcal{K}$.

Remark 3.7.4. *Equivalently, a cubical complex may be viewed as a finite polytopal complex (or regular CW complex) whose cells are combinatorially isomorphic to cubes. In particular, the standard n-cube*

$$Q^n = [0, 1]^n$$

and its k-faces provide the basic local model for cubical complexes.

Example 3.7.5 (Two adjacent unit squares as a cubical complex)**.** Consider the following two elementary 2-cubes in $\mathbb{R}^2$:

$$Q_1 = [0, 1] \times [0, 1], \qquad Q_2 = [1, 2] \times [0, 1].$$

These are two unit squares sharing the common vertical edge

$$Q_1 \cap Q_2 = \{1\} \times [0, 1].$$

Let $\mathcal{K}$ be the collection consisting of Q_1, Q_2, and all of their faces; that is, $\mathcal{K}$ contains the two 2-cubes, all of their 1-dimensional faces, and all of their 0-dimensional faces.

Then $\mathcal{K}$ is a cubical complex. Indeed:

1. every face of Q_1 and Q_2 belongs to $\mathcal{K}$, so $\mathcal{K}$ is closed under taking faces;
2. the intersection of any two cubes in $\mathcal{K}$ is either empty or a face of both cubes. In particular,

$$Q_1 \cap Q_2 = \{1\} \times [0, 1]$$

is a 1-dimensional face of both Q_1 and Q_2.

Therefore $\mathcal{K}$ is a 2-dimensional cubical complex. Its underlying space is

$$|\mathcal{K}| = [0, 2] \times [0, 1].$$

An illustration is given in Fig. 3.6.

3.8 Path Complex

A path complex is a collection of vertex sequences closed under endpoint truncation, generalizing simplicial complexes and directed graphs homologically [206, 207, 208]. In addition to this concept, other related notions are also known, for example, the Flag Complex[209, 210] and the Directed Flag Complex[211, 212, 213].

Definition 3.8.1 (Elementary path)**.** Let V be a finite nonempty set. For an integer $n \geq 0$, an *elementary n-path* on V is a finite sequence

$$i_0 i_1 \cdots i_n \qquad (i_k \in V).$$

For $n = -1$, we denote by e the empty path.

Definition 3.8.2 (Path complex)**.** Let V be a finite nonempty set. A *path complex* on V is a family

$$P = \{P_n\}_{n \geq -1},$$

where each P_n is a set of elementary n-paths on V, satisfying:

1. $P_{-1} = \{e\}$;
2. if
$$i_0 i_1 \cdots i_n \in P_n \qquad (n \geq 0),$$
then both truncated paths
$$i_0 i_1 \cdots i_{n-1} \in P_{n-1}, \qquad i_1 i_2 \cdots i_n \in P_{n-1}$$
also belong to the family.

Elements of P_n are called *allowed n-paths.*

Remark 3.8.3. *Thus, a path complex is a collection of elementary paths closed under the two natural truncation operations: deleting the last vertex and deleting the first vertex.*

Example 3.8.4 (A path complex generated by a directed graph)**.** Let

$$V = \{a, b, c, d\},$$

and consider the directed graph

$$G = (V, E),$$

where

$$E = \{(a, b), (a, c), (b, c), (b, d), (c, d)\}.$$

Equivalently, the directed edges are

$$a \to b, \qquad a \to c, \qquad b \to c, \qquad b \to d, \qquad c \to d.$$

Let $P(G)$ denote the path complex generated by G. By definition, an elementary path

$$i_0 i_1 \cdots i_n$$

belongs to $P_n(G)$ if and only if each consecutive pair forms a directed edge of G, that is,

$$(i_{k-1}, i_k) \in E \qquad (k = 1, \ldots, n).$$

Hence:

$$P_0(G) = \{a, b, c, d\},$$
$$P_1(G) = \{ab, ac, bc, bd, cd\},$$
$$P_2(G) = \{abc, abd, acd, bcd\},$$

and

$$P_3(G) = \{abcd\}.$$

There are no allowed paths of length > 3 in this example.

Indeed,

$$abcd \in P_3(G)$$

because

$$a \to b, \qquad b \to c, \qquad c \to d$$

are all edges of G. Moreover, its truncations

$$abc, \quad bcd \in P_2(G),$$

and the truncations of these 2-paths again belong to $P_1(G)$. Therefore the truncation axiom is satisfied.

An illustration of this path complex is shown in Fig. 3.7.

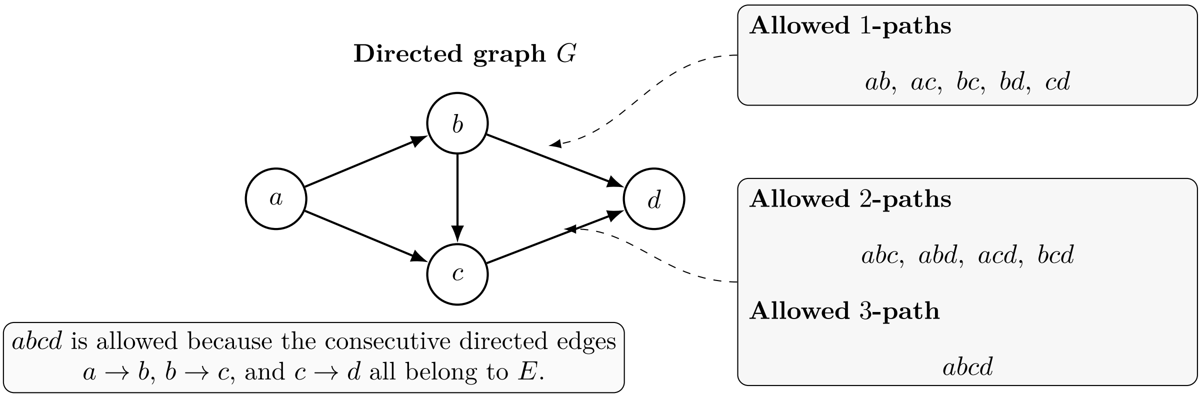


Figure 3.7: A directed graph generating a path complex. Allowed higher-order paths are determined by directed consecutive edges.

3.9 Cellular Sheaf

A cellular sheaf assigns data to cells and restriction maps to incidences, encoding local global consistency on graphs and complexes [214, 215, 216].

Definition 3.9.1 (Cellular sheaf)**.** Let X be a finite regular cell complex, and let P_X denote its face poset, viewed as a category in which there is a unique morphism

$$\sigma \longrightarrow \tau \qquad \text{whenever} \qquad \sigma \leq \tau.$$

Let $\Bbbk$ be a field. A *cellular sheaf of* $\Bbbk$*-vector spaces* on X is a covariant functor

$$\mathcal{F} : P_X \longrightarrow \mathrm{Vect}_{\Bbbk}.$$

Equivalently, a cellular sheaf $\mathcal{F}$ consists of:

1. a $\Bbbk$-vector space $\mathcal{F}(\sigma)$ for each cell $\sigma \in X$, called the *stalk* of $\mathcal{F}$ over σ;
2. for each incidence relation $\sigma \leq \tau$, a linear map

$$\mathcal{F}_{\sigma,\tau} : \mathcal{F}(\sigma) \to \mathcal{F}(\tau),$$

called the *restriction map*;

such that

$$\mathcal{F}_{\sigma,\sigma} = \mathrm{id}_{\mathcal{F}(\sigma)}$$

for every cell σ, and

$$\mathcal{F}_{\tau,\eta} \circ \mathcal{F}_{\sigma,\tau} = \mathcal{F}_{\sigma,\eta}$$

whenever $\sigma \leq \tau \leq \eta$.

Definition 3.9.2 (Global section)**.** Let $\mathcal{F}$ be a cellular sheaf on X. A *global section* of $\mathcal{F}$ is a family

$$s = (s_\sigma)_{\sigma \in X}, \qquad s_\sigma \in \mathcal{F}(\sigma),$$

such that for every incidence relation $\sigma \leq \tau$,

$$\mathcal{F}_{\sigma,\tau}(s_\sigma) = s_\tau.$$

The vector space of global sections is denoted by

$$\Gamma(X; \mathcal{F}).$$

Example 3.9.3 (A cellular sheaf on a graph)**.** Let X be the 1-dimensional regular cell complex consisting of two vertices

$$u, \ v$$

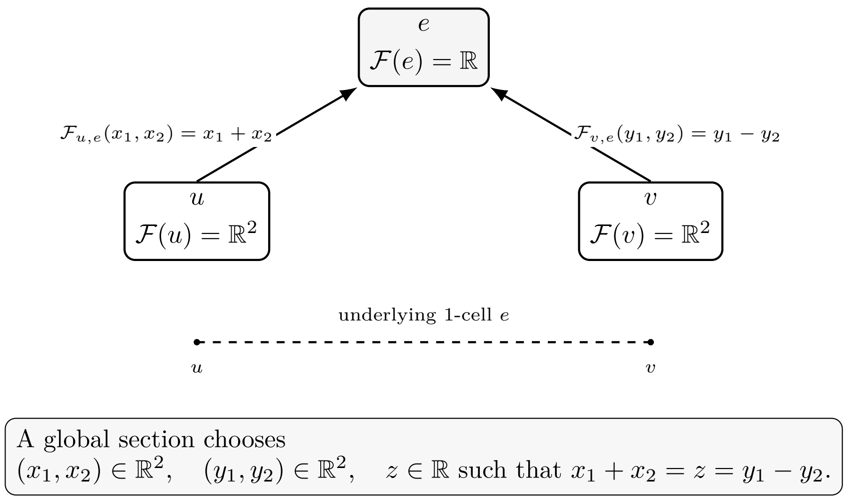


Figure 3.8: A cellular sheaf on a graph with two vertices and one edge. The edge stalk stores a shared scalar compatibility value determined by linear maps from the vertex stalks.

and one edge

$$e$$

with incidence relations

$$u \leq e, \qquad v \leq e.$$

Thus X is the cell complex associated with the interval joining u and v.

Define a cellular sheaf $\mathcal{F} : X \to \mathrm{Vect}_{\mathbb{R}}$ by

$$\mathcal{F}(u) = \mathbb{R}^2, \qquad \mathcal{F}(v) = \mathbb{R}^2, \qquad \mathcal{F}(e) = \mathbb{R}.$$

Define the restriction maps by

$$\mathcal{F}_{u,e}(x_1, x_2) = x_1 + x_2, \qquad \mathcal{F}_{v,e}(y_1, y_2) = y_1 - y_2.$$

Since the only non-identity face relations are $u \leq e$ and $v \leq e$, the functorial conditions are automatically satisfied, and hence $\mathcal{F}$ is a cellular sheaf.

A global section of $\mathcal{F}$ is a triple

$$\big((x_1, x_2), (y_1, y_2), z\big) \in \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}$$

such that

$$z = \mathcal{F}_{u,e}(x_1, x_2) = x_1 + x_2$$

and

$$z = \mathcal{F}_{v,e}(y_1, y_2) = y_1 - y_2.$$

Therefore,

$$\Gamma(X; \mathcal{F}) = \left\{ \big((x_1, x_2), (y_1, y_2), z\big) \in \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R} \;\middle|\; x_1 + x_2 = z, \quad y_1 - y_2 = z \right\}.$$

Eliminating z, we may also write

$$\Gamma(X; \mathcal{F}) \cong \left\{ (x_1, x_2, y_1, y_2) \in \mathbb{R}^4 \;\middle|\; x_1 + x_2 = y_1 - y_2 \right\}.$$

Hence the space of global sections is a 3-dimensional linear subspace of $\mathbb{R}^4$.

An illustration of this sheaf is given in Fig. 3.8.

3.10 Meta Simplicial Complex

We introduce a higher-order structure whose vertices are themselves simplicial complexes. This may be viewed as a *simplicial complex of simplicial complexes*.

Definition 3.10.1 (Universe of finite abstract simplicial complexes)**.** Let X be a finite nonempty set. Define

$$\mathrm{Simp}(X) := \left\{ K \subseteq \mathcal{P}(X) \setminus \{\varnothing\} \;\middle|\; \begin{array}{l} K \neq \varnothing, \text{ and whenever } \sigma \in K \\ \text{and } \varnothing \neq \tau \subseteq \sigma, \text{ then } \tau \in K \end{array} \right\}.$$

Thus $\mathrm{Simp}(X)$ is the family of all finite abstract simplicial complexes on the base set X, where we use the convention that simplices are nonempty subsets of X.

Definition 3.10.2 (Meta simplicial complex)**.** Let X be a finite nonempty set, and let

$$\mathcal{V} \subseteq \mathrm{Simp}(X)$$

be a finite nonempty set of simplicial complexes on X. A *Meta Simplicial Complex* over X is a pair

$$\mathfrak{M} = (\mathcal{V}, \mathcal{K}),$$

where

$$\mathcal{K} \subseteq \mathcal{P}(\mathcal{V}) \setminus \{\varnothing\}$$

satisfies the downward-closure condition:

$$A \in \mathcal{K},\ \varnothing \neq B \subseteq A \quad \Longrightarrow \quad B \in \mathcal{K}.$$

The elements of $\mathcal{V}$ are called *meta-vertices*, and each meta-vertex is itself an abstract simplicial complex on X. The elements of $\mathcal{K}$ are called *meta-simplices*. If $A \in \mathcal{K}$ and $|A| = r + 1$, then A is called an r*-dimensional meta-simplex*. The dimension of $\mathfrak{M}$ is defined by

$$\dim(\mathfrak{M}) := \max\{\, |A| - 1 \mid A \in \mathcal{K} \,\}.$$

Remark 3.10.3. *A Meta Simplicial Complex is an ordinary abstract simplicial complex whose vertices are not atomic objects, but rather simplicial complexes. Hence it is naturally interpreted as a* simplicial complex of simplicial complexes.

Example 3.10.4 (A concrete Meta Simplicial Complex)**.** Let

$$X = \{a, b, c, d\}.$$

Define three simplicial complexes on X:

$$K_1 = \{\{a\}, \{b\}, \{a, b\}\},$$

$$K_2 = \{\{b\}, \{c\}, \{b, c\}\},$$

$$K_3 = \{\{c\}, \{d\}, \{c, d\}\}.$$

Each K_i is an abstract simplicial complex on X, so

$$K_1, K_2, K_3 \in \mathrm{Simp}(X).$$

Let

$$\mathcal{V} = \{K_1, K_2, K_3\}.$$

Define

$$\mathcal{K} = \Big\{\{K_1\}, \{K_2\}, \{K_3\}, \{K_1, K_2\}, \{K_2, K_3\}, \{K_1, K_2, K_3\}\Big\}.$$

Then $\mathcal{K} \subseteq \mathcal{P}(\mathcal{V}) \setminus \{\varnothing\}$, and $\mathcal{K}$ is downward closed with respect to nonempty inclusion. Hence

$$\mathfrak{M} = (\mathcal{V}, \mathcal{K})$$

is a Meta Simplicial Complex over X. In this example, $\{K_1, K_2, K_3\}$ is a 2-dimensional meta-simplex.

Theorem 3.10.5 (Well-definedness of Meta Simplicial Complexes)**.** *Let X be a finite nonempty set. Then:*

1. $\mathrm{Simp}(X)$ *is a well-defined finite set;*

2. *if* $\mathcal{V} \subseteq \mathrm{Simp}(X)$ *is finite and nonempty, and*

$$\mathcal{K} \subseteq \mathcal{P}(\mathcal{V}) \setminus \{\varnothing\}$$

is downward closed under nonempty inclusion, then $\mathcal{K}$ *is an abstract simplicial complex on the vertex set* $\mathcal{V}$*;*

3. *consequently, every pair* $\mathfrak{M} = (\mathcal{V}, \mathcal{K})$ *satisfying Definition 3.10.2 is a well-defined mathematical object.*

Proof. We prove the assertions one by one.

(1) $\mathrm{Simp}(X)$ **is a well-defined finite set.** Since X is finite, its power set $\mathcal{P}(X)$ is finite. Hence

$$\mathcal{P}(X) \setminus \{\varnothing\}$$

is finite, and therefore

$$\mathcal{P}\big(\mathcal{P}(X) \setminus \{\varnothing\}\big)$$

is also finite. By Definition 3.10.1, $\mathrm{Simp}(X)$ is the subclass of

$$\mathcal{P}\big(\mathcal{P}(X) \setminus \{\varnothing\}\big)$$

consisting of those families that satisfy the simplicial downward-closure condition. Thus $\mathrm{Simp}(X)$ is a well-defined subset of a finite set, and hence it is itself finite.

(2) $\mathcal{K}$ **is an abstract simplicial complex on** $\mathcal{V}$**.** By assumption,

$$\mathcal{K} \subseteq \mathcal{P}(\mathcal{V}) \setminus \{\varnothing\}.$$

Thus every element of $\mathcal{K}$ is a nonempty finite subset of $\mathcal{V}$. Moreover, $\mathcal{K}$ is assumed to be downward closed under nonempty inclusion: if $A \in \mathcal{K}$ and $\varnothing \neq B \subseteq A$, then $B \in \mathcal{K}$. This is exactly the defining axiom of an abstract simplicial complex when simplices are taken to be nonempty subsets. Therefore $\mathcal{K}$ is an abstract simplicial complex on vertex set $\mathcal{V}$.

(3) Well-definedness of $\mathfrak{M} = (\mathcal{V}, \mathcal{K})$**.** By part (1), the meta-vertex set $\mathcal{V} \subseteq \mathrm{Simp}(X)$ is drawn from a well-defined finite ambient set. By part (2), the family $\mathcal{K}$ is an abstract simplicial complex on $\mathcal{V}$. Hence the pair

$$\mathfrak{M} = (\mathcal{V}, \mathcal{K})$$

is well-defined. □

Proposition 3.10.6 (Ordinary simplicial complexes are recovered as a special case)**.** *Let* X *be a finite nonempty set, and let* $\mathcal{V} \subseteq \mathrm{Simp}(X)$ *be any finite nonempty set. Then every abstract simplicial complex on the vertex set* $\mathcal{V}$ *determines a Meta Simplicial Complex over* X*.*

Proof. Let $\mathcal{K} \subseteq \mathcal{P}(\mathcal{V}) \setminus \{\varnothing\}$ be any abstract simplicial complex on $\mathcal{V}$. By definition, $\mathcal{K}$ is downward closed under nonempty inclusion. Therefore the pair $(\mathcal{V}, \mathcal{K})$ satisfies Definition 3.10.2, and hence is a Meta Simplicial Complex over X. □

3.11 Simplicial SuperHypercomplex

A superhypercomplex is a simplicial complex whose vertices are nested-set supervertices, capturing higher-order incidence across hierarchical elements.

Definition 3.11.1 (n-SuperHypercomplex)**.** Let V_0 be a nonempty base set and fix an integer $n \in \mathbb{N}_0$. Define the iterated powersets recursively by

$$\mathcal{P}^0(V_0) := V_0, \qquad \mathcal{P}^{k+1}(V_0) := \mathcal{P}\big(\mathcal{P}^k(V_0)\big) \quad (k \geq 0).$$

Choose a set of *n-supervertices*

$$V \subseteq \mathcal{P}^n(V_0).$$

An *n-SuperHypercomplex* on V_0 (with supervertex set V) is a pair

$$\mathrm{SuHyC}^{(n)} = (V, \mathcal{K}),$$

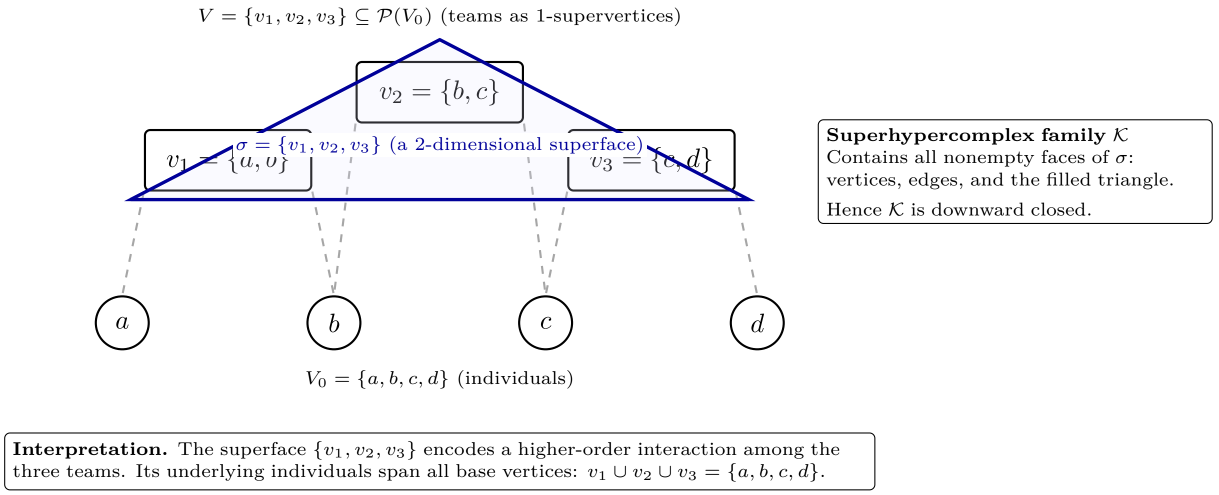


Figure 3.9: A 1-SuperHypercomplex built from teams of individuals (Example 3.11.2). The filled super-triangle represents the 2-dimensional superface $\sigma = \{v_1, v_2, v_3\}$, and dashed links indicate team membership at the base level.

where $\mathcal{K} \subseteq \mathcal{P}(V) \setminus \{\emptyset\}$ is a family of nonempty finite subsets of V satisfying the *downward-closure* (face) condition:

$$\sigma \in \mathcal{K},\ \emptyset \neq \tau \subseteq \sigma \implies \tau \in \mathcal{K}.$$

Elements of $\mathcal{K}$ are called *n-superfaces* (or *super-simplices*). If $|\sigma| = k + 1$, then σ is a *k-dimensional superface* and we set $\dim(\sigma) := k$. The *dimension* of $\mathrm{SuHyC}^{(n)}$ is

$$\dim\big(\mathrm{SuHyC}^{(n)}\big) := \sup\{\dim(\sigma) \mid \sigma \in \mathcal{K}\}.$$

Example 3.11.2 (A 1-SuperHypercomplex built from teams of individuals)**.** Let the base set be

$$V_0 = \{a, b, c, d\},$$

and take $n = 1$, so $\mathcal{P}^1(V_0) = \mathcal{P}(V_0)$. Interpret 1-supervertices as *teams* (nonempty subsets of V_0), and choose

$$V = \{\, v_1 = \{a, b\},\ v_2 = \{b, c\},\ v_3 = \{c, d\} \,\} \subseteq \mathcal{P}(V_0).$$

Define a family $\mathcal{K} \subseteq \mathcal{P}(V) \setminus \{\emptyset\}$ by taking one 2-dimensional superface (a super-triangle) and all of its nonempty faces:

$$\sigma = \{v_1, v_2, v_3\} \in \mathcal{P}(V).$$

Set

$$\mathcal{K} = \Big\{\{v_1\}, \{v_2\}, \{v_3\}, \{v_1, v_2\}, \{v_1, v_3\}, \{v_2, v_3\}, \{v_1, v_2, v_3\}\Big\}.$$

By construction, $\mathcal{K}$ is downward closed: whenever τ is a nonempty subset of σ, we have $\tau \in \mathcal{K}$. Hence

$$\mathrm{SuHyC}^{(1)} = (V, \mathcal{K})$$

is a 1-SuperHypercomplex in the sense of Definition 3.11.1. Its dimension is 2, since it contains the 2-dimensional superface $\{v_1, v_2, v_3\}$.

The superface $\{v_1, v_2, v_3\}$ encodes a higher-order incidence among the three teams. At the base level, the union of underlying individuals involved is

$$v_1 \cup v_2 \cup v_3 = \{a, b, c, d\},$$

so the super-triangle can be viewed as a "team-of-teams" interaction that collectively spans all four individuals. An overview diagram of this example is provided in Fig. 3.9.

4 Factorization, constraint, layered, temporal, and tensor-based family

Factorization, constraint, layered, temporal, and tensor-based families model higher-order systems through decomposed interactions, coding constraints, coupled layers, time-varying relations, and tensor representations, enabling efficient analysis of complex multiway dependencies and patterns. For reference, the factorization, constraint, layered, temporal, and tensor-based higher-order structures treated in this book are listed in Table 4.1.

Table 4.1: Factorization, constraint, layered, temporal, and tensor-based higher-order structures treated in this book.

Concept	**Concise description**
Factor graph	A bipartite graph linking variables and factors, representing how a global function decomposes into local interaction terms.
Tanner graph	A bipartite graph linking variable nodes and parity-check nodes, encoding constraint structure in linear codes.
Tanner Hypergraph	A hypergraph form of parity-check structure, where each check induces a hyperedge over its participating variables.
Tanner SuperHyperGraph	A superhypergraph extension of Tanner-type coding structure, allowing grouped or hierarchical variable relations through supervertices.
Multilayer network	A network with node–layer states and intra-/interlayer edges, modeling multiple coupled interaction contexts or modalities.
Temporal network	A time-indexed network whose edges occur at specific times or intervals, capturing evolving interaction patterns.
MultiDimensional Graph (Cartesian-product graph)	A graph built as a Cartesian product of factor graphs, encoding multi-axis adjacency through coordinatewise changes.
Adjacency-Tensor Network (ATN)	A tensor-based higher-order network model using adjacency tensors to represent multiway interactions beyond pairwise incidence.

4.1 Factor graph

A factor graph is a bipartite graph linking variables to factors, representing how a global function factorizes into local terms [217, 218, 219]. A factor graph represents higher-order networks by introducing factor nodes for multiway interactions, converting hyperedge constraints into bipartite edges linking variables to interaction factors.

Definition 4.1.1 (Factor graph)**.** Let $X = \{x_1, \dots, x_n\}$ be a set of variables and let $\mathcal{F} = \{f_1, \dots, f_m\}$ be a set of factors. A *factor graph* is a bipartite graph

$$\mathcal{G}_F = (X \sqcup \mathcal{F},\ E),$$

where an edge connects a variable node x_j to a factor node f_i if and only if f_i depends on x_j. Equivalently, each factor f_i has a *scope* (neighbor set)

$$\mathrm{scope}(f_i) := \{x_j \in X \mid \{x_j, f_i\} \in E\} \subseteq X.$$

In probabilistic modeling, a function $F : X_1 \times \cdots \times X_n \to \mathbb{R}_{\geq 0}$ (often a joint density or unnormalized potential) is said to *factorize over* $\mathcal{G}_F$ if

$$F(x_1, \ldots, x_n) = \prod_{i=1}^{m} f_i\big((x_j)_{x_j \in \mathrm{scope}(f_i)}\big).$$

Example 4.1.2 (A factor graph for a simple Markov chain)**.** Let x_1, x_2, x_3 be random variables taking values in finite sets X_1, X_2, X_3. Consider a chain-structured factorization of a nonnegative function

$$F : X_1 \times X_2 \times X_3 \to \mathbb{R}_{\geq 0}$$

of the form

$$F(x_1, x_2, x_3) = f_{12}(x_1, x_2)\, f_{23}(x_2, x_3),$$

where

$$f_{12} : X_1 \times X_2 \to \mathbb{R}_{\geq 0}, \qquad f_{23} : X_2 \times X_3 \to \mathbb{R}_{\geq 0}$$

are pairwise factors.

Define the variable-node set and factor-node set by

$$X = \{x_1, x_2, x_3\}, \qquad \mathcal{F} = \{f_{12}, f_{23}\}.$$

The corresponding factor graph is the bipartite graph

$$\mathcal{G}_F = (X \sqcup \mathcal{F},\ E),$$

with edge set

$$E = \big\{\{x_1, f_{12}\}, \{x_2, f_{12}\}, \{x_2, f_{23}\}, \{x_3, f_{23}\}\big\}.$$

Equivalently, the scopes are

$$\mathrm{scope}(f_{12}) = \{x_1, x_2\}, \qquad \mathrm{scope}(f_{23}) = \{x_2, x_3\}.$$

Thus F factorizes over $\mathcal{G}_F$ exactly as in Definition 4.1.1.

4.2 Tanner graph

A Tanner graph is a bipartite graph connecting variable nodes to parity-check nodes, representing nonzero entries of a code's matrix [220, 221, 222]. A Tanner graph models higher-order networks by bipartite variable and constraint nodes; each constraint encodes multiway relations, with edges linking involved variables for inference efficiently.

Definition 4.2.1 (Tanner graph)**.** Let $\mathbb{F}$ be a finite field and let

$$H = (h_{ij}) \in \mathbb{F}^{m \times n}$$

be a parity-check matrix of a linear code. Let $X = \{x_1, \ldots, x_n\}$ be the set of *variable nodes* and $C = \{c_1, \ldots, c_m\}$ be the set of *check nodes*. The *Tanner graph* of H is the bipartite graph

$$\mathrm{TG}(H) = (X \sqcup C,\ E)$$

with

$$E := \big\{\{x_j, c_i\} \ \big|\ h_{ij} \neq 0\big\}.$$

In the nonbinary case one often equips each edge $\{x_j, c_i\} \in E$ with the label $\lambda(\{x_j, c_i\}) = h_{ij} \in \mathbb{F}^{\times}$.

Example 4.2.2 (A Tanner graph for a small binary parity-check code)**.** Let $\mathbb{F} = \mathbb{F}_2$ and consider the parity-check matrix

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \in \mathbb{F}_2^{2 \times 4}.$$

Thus $m = 2$ and $n = 4$. Let the variable nodes and check nodes be

$$X = \{x_1, x_2, x_3, x_4\}, \qquad C = \{c_1, c_2\}.$$

By Definition 4.2.1, the edge set of the Tanner graph $\mathrm{TG}(H)$ is determined by the nonzero entries of H:

$$E = \big\{\{x_1, c_1\}, \{x_2, c_1\}, \{x_4, c_1\}, \{x_2, c_2\}, \{x_3, c_2\}, \{x_4, c_2\}\big\}.$$

Equivalently, the first check node c_1 is adjacent to x_1, x_2, x_4 and the second check node c_2 is adjacent to x_2, x_3, x_4. Hence $\mathrm{TG}(H)$ is a bipartite graph that encodes the incidence pattern of the parity-check equations represented by H.

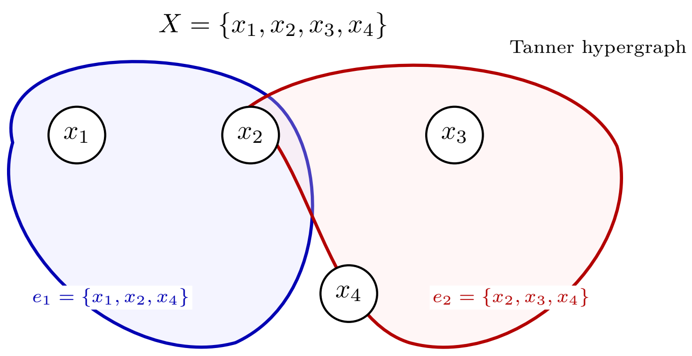


Figure 4.1: A Tanner hypergraph for the parity-check matrix in Example 4.3.3. The two hyperedges correspond to the row supports $\mathrm{supp}(H_1) = \{1, 2, 4\}$ and $\mathrm{supp}(H_2) = \{2, 3, 4\}$.

4.3 Tanner Hypergraph

A Tanner Hypergraph models coding constraints by linking variable nodes and check hyperedges, representing multiway parity relations in error-correcting codes. For each row $i \in \{1, \ldots, m\}$, define its support by

$$\mathrm{supp}(H_i) := \{\, j \in \{1, \ldots, n\} \;\big|\; h_{ij} \neq 0 \,\}.$$

Definition 4.3.1 (Tanner hypergraph)**.** The *Tanner hypergraph* of H is the hypergraph

$$\mathrm{TH}(H) = (X,\ \mathcal{E}),$$

where

$$\mathcal{E} := \{e_1, \ldots, e_m\}, \qquad e_i := \{\, x_j \in X \mid j \in \mathrm{supp}(H_i) \,\} \subseteq X.$$

In the nonbinary case, one may record coefficients by an incidence-weight map

$$w_i : e_i \to \mathbb{F}^{\times}, \qquad w_i(x_j) := h_{ij},$$

so that $\mathrm{TH}(H)$ becomes a weighted (or labeled) hypergraph.

Remark 4.3.2 (Incidence graph)**.** *The incidence bipartite graph of* $\mathrm{TH}(H)$ *(with coefficient labels, when present) coincides with the Tanner graph* $\mathrm{TG}(H)$*.*

Example 4.3.3 (A Tanner hypergraph for a small binary parity-check matrix)**.** Let $\mathbb{F} = \mathbb{F}_2$ and consider the parity-check matrix

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \in \mathbb{F}_2^{2\times 4}.$$

Let $X = \{x_1, x_2, x_3, x_4\}$ be the variable set. The row supports are

$$\mathrm{supp}(H_1) = \{1, 2, 4\}, \qquad \mathrm{supp}(H_2) = \{2, 3, 4\}.$$

Hence the Tanner hypergraph $\mathrm{TH}(H) = (X, \mathcal{E})$ has hyperedge family

$$\mathcal{E} = \{e_1, e_2\}, \qquad e_1 = \{x_1, x_2, x_4\}, \qquad e_2 = \{x_2, x_3, x_4\}.$$

In particular, e_1 encodes the first parity-check equation involving variables x_1, x_2, x_4, and e_2 encodes the second parity-check equation involving x_2, x_3, x_4. An overview diagram of this example is provided in Fig. 4.1.

4.4 Tanner SuperHyperGraph

A Tanner SuperHyperGraph extends Tanner hypergraphs by allowing hierarchical supervertices and superhyperedges, modeling nested coding constraints, multilevel parity relations, and higher-order dependency structures in codes efficiently. To interpret a nested set as the collection of underlying variables it represents, define recursively:

$$\mathrm{flat}_0(x) := \{x\} \subseteq V_0 \quad (x \in V_0),$$

and for $k \geq 1$ and $A \in \mathcal{P}^k(V_0) = \mathcal{P}(\mathcal{P}^{k-1}(V_0))$,

$$\mathrm{flat}_k(A) := \bigcup_{B \in A} \mathrm{flat}_{k-1}(B) \subseteq V_0.$$

In particular, $\mathrm{flat}_k(A)$ is always a (possibly empty) subset of V_0.

Definition 4.4.1 (Tanner n-superhypergraph)**.** Fix $n \in \mathbb{N}$. Let $V \subseteq \mathcal{P}^n(V_0)$ be a set of *n-supervertices.* For each check $i \in \{1, \dots, m\}$ choose a nonempty *superhyperedge*

$$E_i \in \mathcal{P}(V) \setminus \{\emptyset\}.$$

Put $\mathcal{E} := \{E_1, \dots, E_m\}$ and define

$$\mathrm{TSH}^{(n)}(H) := (V, \mathcal{E}).$$

We call $\mathrm{TSH}^{(n)}(H)$ a *Tanner n-superhypergraph* (for H) if for every i,

$$\bigcup_{v \in E_i} \mathrm{flat}_n(v) \;=\; \{\, x_j \mid j \in \mathrm{supp}(H_i) \,\}. \tag{T}$$

Example 4.4.2 (A Tanner 1-superhypergraph obtained by grouping variables)**.** Use the same base variable set

$$V_0 = X = \{x_1, x_2, x_3, x_4\},$$

and the same parity-check matrix H as in Example 4.3.3. Take $n = 1$, so 1-supervertices are nonempty subsets of V_0.

Define the 1-supervertex set (variable groups)

$$V = \{v_A = \{x_1, x_2\},\ v_B = \{x_2, x_3\},\ v_C = \{x_4\}\} \subseteq \mathcal{P}^1(V_0) = \mathcal{P}(V_0).$$

Define two superhyperedges (one per check equation) by

$$E_1 = \{v_A, v_C\}, \qquad E_2 = \{v_B, v_C\}, \qquad \mathcal{E} = \{E_1, E_2\}.$$

Since $n = 1$, we have $\mathrm{flat}_1(v) = v$ for every $v \in V$. Therefore,

$$\bigcup_{v \in E_1} \mathrm{flat}_1(v) = v_A \cup v_C = \{x_1, x_2, x_4\} = \{x_j \mid j \in \mathrm{supp}(H_1)\},$$

and

$$\bigcup_{v \in E_2} \mathrm{flat}_1(v) = v_B \cup v_C = \{x_2, x_3, x_4\} = \{x_j \mid j \in \mathrm{supp}(H_2)\}.$$

Hence the condition (T) holds for $i = 1, 2$, and thus

$$\mathrm{TSH}^{(1)}(H) = (V, \mathcal{E})$$

is a Tanner 1-superhypergraph in the sense of Definition 4.4.1. Intuitively, the first check is represented by the grouped variable block $\{x_1, x_2\}$ together with $\{x_4\}$, and the second check by the block $\{x_2, x_3\}$ together with $\{x_4\}$.

4.5 Multilayer network

A multilayer network represents nodes across multiple layers, using node–layer state nodes with intra- and interlayer edges modeling diverse relations [223, 224, 225].

Definition 4.5.1 (Multilayer network)**.** A *multilayer network* is a tuple

$$\mathcal{M} = (V,\, L_1, \dots, L_p,\, V_M,\, E_M),$$

where V is a set of physical entities (nodes), each L_i is a finite set of *elementary layer labels*, the set of *layers* is the Cartesian product $L := L_1 \times \cdots \times L_p$, and

$$V_M \subseteq V \times L$$

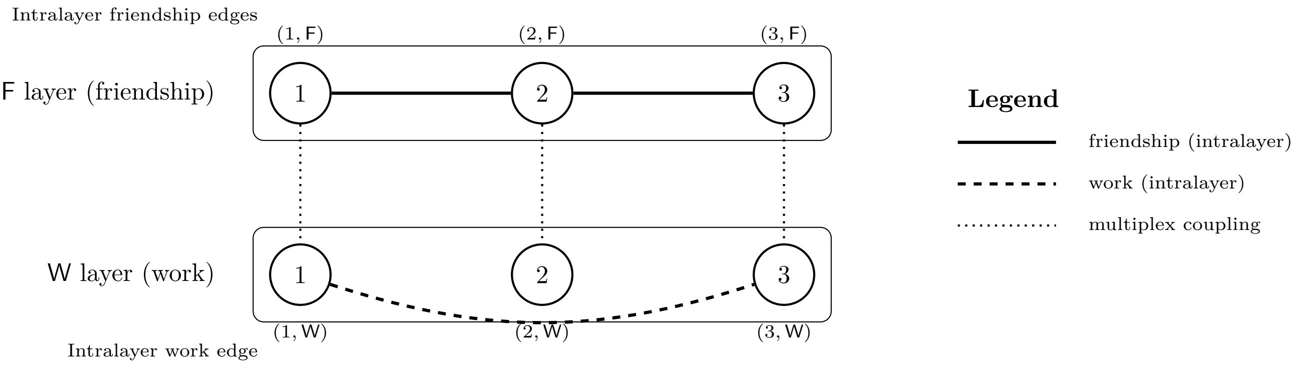


Figure 4.2: A simple multiplex multilayer network with two layers: friendship (F) and work collaboration (W). Dotted vertical edges connect the same physical node across layers, hence the network is multiplex.

is the set of *state nodes* (node–layer tuples). The edge set is

$$E_M \subseteq V_M \times V_M$$

(undirected variants use $E_M \subseteq \binom{V_M}{2}$). Edges with endpoints in the same layer are called *intralayer edges*, and edges between different layers are called *interlayer edges*. A *multiplex network* is a multilayer network for which interlayer edges are allowed only between state nodes corresponding to the same physical node, i.e.

$$\big((v,\ell),(v',\ell')\big) \in E_M \ \text{ and } \ \ell \neq \ell' \quad \Longrightarrow \quad v = v'.$$

Example 4.5.2 (A simple multiplex multilayer network: friendship vs. collaboration)**.** Let the set of physical nodes be

$$V = \{1, 2, 3\}.$$

Take a single aspect ($p = 1$) with two layer labels

$$L_1 = \{\mathsf{F}, \mathsf{W}\},$$

where F denotes a friendship layer and W denotes a work-collaboration layer. Thus the layer set is $L = L_1$ and the state-node set is

$$V_M = V \times L = \{(1,\mathsf{F}), (2,\mathsf{F}), (3,\mathsf{F}), (1,\mathsf{W}), (2,\mathsf{W}), (3,\mathsf{W})\}.$$

Define an edge set $E_M \subseteq \binom{V_M}{2}$ (undirected) by specifying:

- *Intralayer friendship edges:*
$$\{(1,\mathsf{F}), (2,\mathsf{F})\}, \quad \{(2,\mathsf{F}), (3,\mathsf{F})\};$$
- *Intralayer work edges:*
$$\{(1,\mathsf{W}), (3,\mathsf{W})\};$$
- *Interlayer coupling (multiplex) edges:*
$$\{(1,\mathsf{F}), (1,\mathsf{W})\}, \quad \{(2,\mathsf{F}), (2,\mathsf{W})\}, \quad \{(3,\mathsf{F}), (3,\mathsf{W})\}.$$

Then

$$\mathcal{M} = (V, L_1, V_M, E_M)$$

is a multilayer network in the sense of Definition 4.5.1. Moreover, it is a *multiplex* network because every interlayer edge connects state nodes of the same physical node. An overview diagram of this example is provided in Fig. 4.2.

We consider further extensions. An iterated multilayer network has nodes that are multilayer networks themselves, defined recursively to depth r, with edges between these network-objects across layers.

Definition 4.5.3 (Iterated Multilayer Network (depth r))**.** Fix a nonempty set U of *atomic entities* (base-level physical nodes). Define, for each $r \in \mathbb{N}_0$, a class $\mathsf{IMLN}_r(U)$ of *iterated multilayer networks of depth* r recursively as follows.

1. **(Depth** 0**)** $\mathsf{IMLN}_0(U)$ consists of all multilayer networks

$$\mathcal{M} = (V,\, L_1, \ldots, L_p,\, V_M,\, E_M)$$

in the sense of Definition 4.5.1, whose physical node set satisfies $V \subseteq U$.

2. **(Depth** $r+1$**)** Assuming $\mathsf{IMLN}_r(U)$ is defined, a multilayer network

$$\mathcal{M} = (V,\, L_1, \ldots, L_p,\, V_M,\, E_M)$$

belongs to $\mathsf{IMLN}_{r+1}(U)$ if it satisfies Definition 4.5.1 and, in addition, its physical node set is a set of depth-r iterated multilayer networks, i.e.

$$V \subseteq \mathsf{IMLN}_r(U).$$

Equivalently, the state-node set is a subset

$$V_M \subseteq V \times (L_1 \times \cdots \times L_p),$$

so a state node has the form $(\mathcal{N}, \ell)$ where $\mathcal{N} \in \mathsf{IMLN}_r(U)$ is itself a (multilayer) network-object (a *meta-node*) and ℓ is a layer label at the current level.

Any $\mathcal{M} \in \mathsf{IMLN}_r(U)$ is called an *Iterated Multilayer Network of depth r*.

Example 4.5.4 (A depth-1 iterated multilayer network: teams as meta-nodes)**.** Let the atomic entity set be

$$U = \{a, b, c, d\}.$$

First build two depth-0 multilayer networks (ordinary multilayer networks) on subsets of U:

Team A network. Let $V^A = \{a, b\} \subseteq U$ and $L_1^A = \{\mathsf{Chat}, \mathsf{Code}\}$. Set $V_M^A = V^A \times L_1^A$ and define $E_M^A \subseteq \binom{V_M^A}{2}$ by

$$\{(a, \mathsf{Chat}), (b, \mathsf{Chat})\} \in E_M^A, \qquad \{(a, \mathsf{Code}), (b, \mathsf{Code})\} \in E_M^A,$$

together with multiplex couplings $\{(a, \mathsf{Chat}), (a, \mathsf{Code})\}$ and $\{(b, \mathsf{Chat}), (b, \mathsf{Code})\}$. Denote this depth-0 multilayer network by $\mathcal{N}_A \in \mathsf{IMLN}_0(U)$.

Team B network. Similarly, let $V^B = \{c, d\} \subseteq U$ with the same layer set $L_1^B = \{\mathsf{Chat}, \mathsf{Code}\}$, and define $\mathcal{N}_B \in \mathsf{IMLN}_0(U)$ analogously.

Top-level (depth-1**) network.** Now form a multilayer network whose physical nodes are the two team-networks:

$$V = \{\mathcal{N}_A, \mathcal{N}_B\} \subseteq \mathsf{IMLN}_0(U).$$

Let the top-level layer set be $L_1 = \{\mathsf{Org}\}$ (one layer), so $V_M = V \times L_1$ consists of two state nodes $(\mathcal{N}_A, \mathsf{Org})$ and $(\mathcal{N}_B, \mathsf{Org})$. Define the top-level edge set by

$$E_M = \big\{\{(\mathcal{N}_A, \mathsf{Org}), (\mathcal{N}_B, \mathsf{Org})\}\big\}.$$

Then

$$\mathcal{M} = (V, L_1, V_M, E_M)$$

is a multilayer network satisfying $V \subseteq \mathsf{IMLN}_0(U)$, hence

$$\mathcal{M} \in \mathsf{IMLN}_1(U).$$

Therefore $\mathcal{M}$ is an *Iterated Multilayer Network of depth* 1 in the sense of Definition 4.5.3, where each physical node is itself a (depth-0) multilayer network.

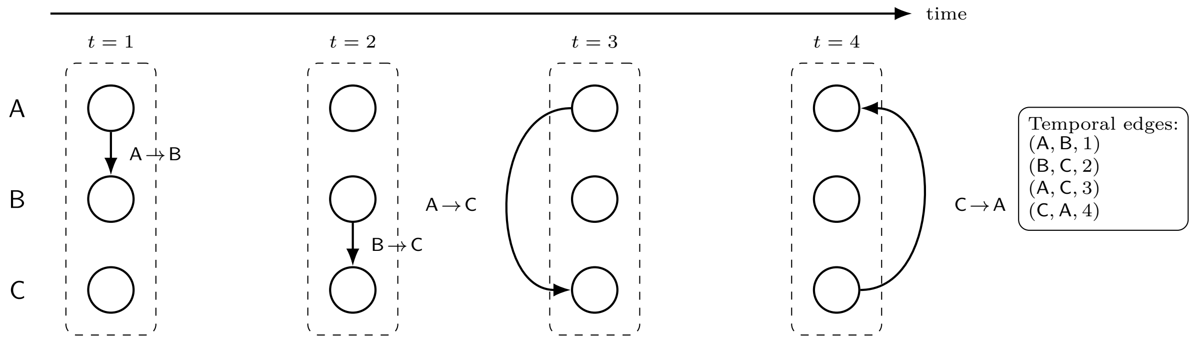


Figure 4.3: A temporal communication network over discrete time $T = \{1, 2, 3, 4\}$. Each dashed box represents one time slice, and the directed edge inside it indicates the message sent at that time.

4.6 Temporal network

A temporal network models time-stamped interactions: edges occur at specific times or intervals, capturing evolving connectivity and causal paths [226, 227, 228].

Definition 4.6.1 (Temporal network (time-varying graph))**.** Let T be a time set (e.g. $T = \mathbb{R}_{\geq 0}$ or $T = \mathbb{Z}$) and let V be a node set. A *temporal network* is a pair

$$\mathcal{G} = (V, E_T),$$

where the *temporal edge set* is

$$E_T \subseteq V \times V \times T.$$

An element $(u, v, t) \in E_T$ means that an interaction (edge) from u to v occurs at time t (undirected variants identify (u, v, t) and (v, u, t)). More generally, one may specify edge activity by an *availability function*

$$\rho : V \times V \times T \to \{0, 1\} \quad \text{(or to } \mathbb{R}_{\geq 0} \text{ for weights)},$$

where $\rho(u, v, t) = 1$ indicates that the edge (u, v) is active at time t.

Example 4.6.2 (A temporal communication network over discrete time)**.** Let the node set be

$$V = \{\mathsf{A}, \mathsf{B}, \mathsf{C}\}$$

and let the time set be the discrete set

$$T = \{1, 2, 3, 4\} \subset \mathbb{Z}.$$

Suppose that at time $t = 1$ user A messages B, at time $t = 2$ user B messages C, at time $t = 3$ user A messages C, and at time $t = 4$ user C replies to A. Define the temporal edge set by

$$E_T = \big\{(\mathsf{A}, \mathsf{B}, 1),\ (\mathsf{B}, \mathsf{C}, 2),\ (\mathsf{A}, \mathsf{C}, 3),\ (\mathsf{C}, \mathsf{A}, 4)\big\} \subseteq V \times V \times T.$$

Then $\mathcal{G} = (V, E_T)$ is a temporal network in the sense of Definition 4.6.1.

Equivalently, one may encode the same data by an availability function $\rho : V \times V \times T \to \{0, 1\}$ defined by $\rho(u, v, t) = 1$ exactly for the above four triples and $\rho(u, v, t) = 0$ otherwise. An overview diagram of this example is provided in Fig. 4.3.

4.7 MultiDimensional Graph (Cartesian-product graph)

A multidimensional graph is the Cartesian product of factor graphs; vertices are tuples, edges change exactly one coordinate per step [229, 230, 231].

Definition 4.7.1 (MultiDimensional Graph (Cartesian-product graph))**.** Let $d \in \mathbb{N}$ and, for each $k \in \{1, \dots, d\}$, let

$$G_k = (V_k, E_k, w_k)$$

be an undirected weighted graph, where V_k is a finite vertex set, $E_k \subseteq \binom{V_k}{2}$, and $w_k : V_k \times V_k \to \mathbb{R}_{\geq 0}$ is symmetric with $w_k(u, v) = 0$ whenever $\{u, v\} \notin E_k$. The *d-dimensional graph* (or *multidimensional graph*) associated with $(G_1, \dots, G_d)$ is the Cartesian product graph

$$G \;=\; G_1 \,\square\, G_2 \,\square\, \cdots \,\square\, G_d \;=\; (V, E, w),$$

defined as follows:

Table 4.2: Difference between a MultiDimensional Graph and a MultiLayer Network (concise view).

Aspect	**MultiDimensional Graph (Def. 4.7.1)**	**MultiLayer Network**
Primitive idea	A *single* graph built as a Cartesian product of factor graphs.	A *collection of layers* (contexts/relations), represented by node–layer state nodes.
Node objects	Tuples $x = (x_1, \ldots, x_d) \in \prod_{k=1}^{d} V_k$.	State nodes (v, ℓ) where v is a physical node and ℓ is a layer label (possibly multi-aspect).
Edge rule	$\{x, y\}$ exists iff x, y differ in exactly one coordinate, along an edge of the corresponding factor graph.	Intralayer edges connect nodes within the same layer; interlayer edges connect nodes across layers (application-defined).
Meaning of "dimension/layer"	"Dimension" d means the number of factor graphs; each coordinate gives a directional axis.	"Layer" indexes different relation types, times, modalities, or contexts; not necessarily a Cartesian axis.
Structural constraint	Strong constraint: global adjacency is *fully determined* by factor graphs (product structure).	Flexible: no product constraint; layer coupling can be arbitrary (e.g. multiplex coupling for the same node).
Typical use	Product-structured domains (e.g. grid-like data, space×time products, separable directional analysis).	Multi-relational / multi-context networks (e.g. multiple interaction types, modalities, snapshots with cross-layer couplings).

1. The vertex set is the Cartesian product
$$V = \prod_{k=1}^{d} V_k.$$

2. Two distinct vertices $x = (x_1, \ldots, x_d)$ and $y = (y_1, \ldots, y_d)$ are adjacent, i.e. $\{x, y\} \in E$, if and only if there exists an index $i \in \{1, \ldots, d\}$ such that
$$\{x_i, y_i\} \in E_i \quad \text{and} \quad x_j = y_j \text{ for all } j \neq i.$$

3. The weight function $w : V \times V \to \mathbb{R}_{\geq 0}$ is given by
$$w(x, y) = \begin{cases} w_i(x_i, y_i), & \text{if } x \text{ and } y \text{ differ only in coordinate } i \text{ and } \{x_i, y_i\} \in E_i, \\ 0, & \text{otherwise.} \end{cases}$$

The graphs $G_1, \ldots, G_d$ are called the *factor graphs*, and the integer d is the *dimension* of G.

For reference, the difference between a MultiDimensional Graph and a MultiLayer Network is summarized in Table 4.2.

We formalize the idea of a *MultiDimensional Graph of MultiDimensional Graphs* by iterating the Cartesian-product construction.

Definition 4.7.2 (Iterated MultiDimensional Graph (IMDG))**.** Fix a class Graph of (undirected) weighted graphs $G = (V, E, w)$ as in Definition 4.7.1. For $r \in \mathbb{N}_0$, define the classes IMDG_r recursively as follows:

1. **(Depth** 0**)** $\mathsf{IMDG}_0 := \mathsf{Graph}$.

2. **(Depth** $r+1$**)** A graph G belongs to IMDG_{r+1} if there exist an integer $d \geq 1$ and graphs $G_1, \ldots, G_d \in \mathsf{IMDG}_r$ such that
$$G \cong G_1 \,\square\, G_2 \,\square\, \cdots \,\square\, G_d,$$
where $\square$ denotes the Cartesian product of graphs (so $V(G) = \prod_{i=1}^{d} V(G_i)$ and adjacency changes exactly one coordinate along an edge of the corresponding factor).

Any $G \in \mathsf{IMDG}_r$ is called an *Iterated MultiDimensional Graph of depth r*.

Definition 4.7.3 (Depth and iterated product specification)**.** For a graph G, define its *iterated multidimensional depth* by

$$\operatorname{depth}(G) := \min\{\, r \in \mathbb{N}_0 \mid G \in \mathsf{IMDG}_r \,\} \quad \text{(if such } r \text{ exists)}.$$

An *iterated product specification* of depth r for G is a rooted decomposition tree whose internal nodes are labeled by Cartesian products and whose leaves are base graphs in IMDG_0, such that evaluating the tree (by taking $\square$ at each internal node) yields a graph isomorphic to G.

Remark 4.7.4 (Flattening)**.** *Because the Cartesian product is associative up to canonical graph isomorphism, any iterated product specification can be* flattened *to a single Cartesian product of all leaf graphs. The iterated specification, however, records a meaningful hierarchical (multi-level) product structure: a "MultiDimensional Graph of MultiDimensional Graphs."*

Example 4.7.5 (An Iterated MultiDimensional Graph of depth 2)**.** Let P_2 denote the path graph on two vertices (a single edge), and let C_4 denote the 4-cycle. Consider the depth-1 multidimensional graph

$$G^{(1)} \;=\; P_2 \,\square\, P_2,$$

which is the 2×2 grid graph (a square) with vertex set

$$V(G^{(1)}) = \{0,1\} \times \{0,1\}$$

and edges between vertices that differ in exactly one coordinate.

Now form another depth-1 multidimensional graph

$$H^{(1)} \;=\; C_4 \,\square\, P_2,$$

whose vertex set is $V(C_4) \times V(P_2)$ and whose adjacency again changes exactly one coordinate.

Finally, define the graph

$$G \;=\; G^{(1)} \,\square\, H^{(1)}.$$

By construction, G is a Cartesian product of two factors $G^{(1)}$ and $H^{(1)}$, each of which is itself a Cartesian product of depth-0 graphs. Hence,

$$G \in \mathsf{IMDG}_2,$$

so G is an Iterated MultiDimensional Graph of depth 2 in the sense of Definition 4.7.2.

Iterated product specification. One iterated product specification for G is given by the rooted decomposition tree

$$\big(P_2 \square P_2\big) \,\square\, \big(C_4 \square P_2\big),$$

whose leaves are $P_2, P_2, C_4, P_2 \in \mathsf{IMDG}_0$ and whose internal nodes apply the Cartesian product. Evaluating this tree yields a graph isomorphic to G, as required by Definition 4.7.3.

4.8 Adjacency-Tensor Network (ATN)

An Adjacency-Tensor Network represents higher-order interactions by tensors of orders 2..K on a vertex set; nonzero entries encode weighted multiway adjacency patterns.

Definition 4.8.1 (Adjacency-Tensor Network (ATN))**.** Let $V = \{1, \ldots, n\}$ be a finite vertex set and fix $K \geq 2$. An *Adjacency-Tensor Network* is a family of real-valued tensors

$$\mathsf{ATN} = \mathbf{A} = \big(A^{(2)}, A^{(3)}, \ldots, A^{(K)}\big), \qquad A^{(k)} \in \mathbb{R}^{n \times \cdots \times n} \;\; (k \text{ factors}),$$

where $A^{(k)}_{i_1 \cdots i_k}$ represents the weight (or indicator) of a k-way interaction among $(i_1, \ldots, i_k) \in V^k$. In the undirected k-uniform case one typically requires symmetry:

$$A^{(k)}_{i_{\pi(1)} \cdots i_{\pi(k)}} = A^{(k)}_{i_1 \cdots i_k} \quad \text{for all } \pi \in S_k.$$

Example 4.8.2 (An ATN with pairwise and triple interactions on $V = \{1, 2, 3\}$)**.** Let $V = \{1, 2, 3\}$, so $n = 3$, and take $K = 3$. Define an Adjacency-Tensor Network

$$\mathbf{A} = \big(A^{(2)}, A^{(3)}\big)$$

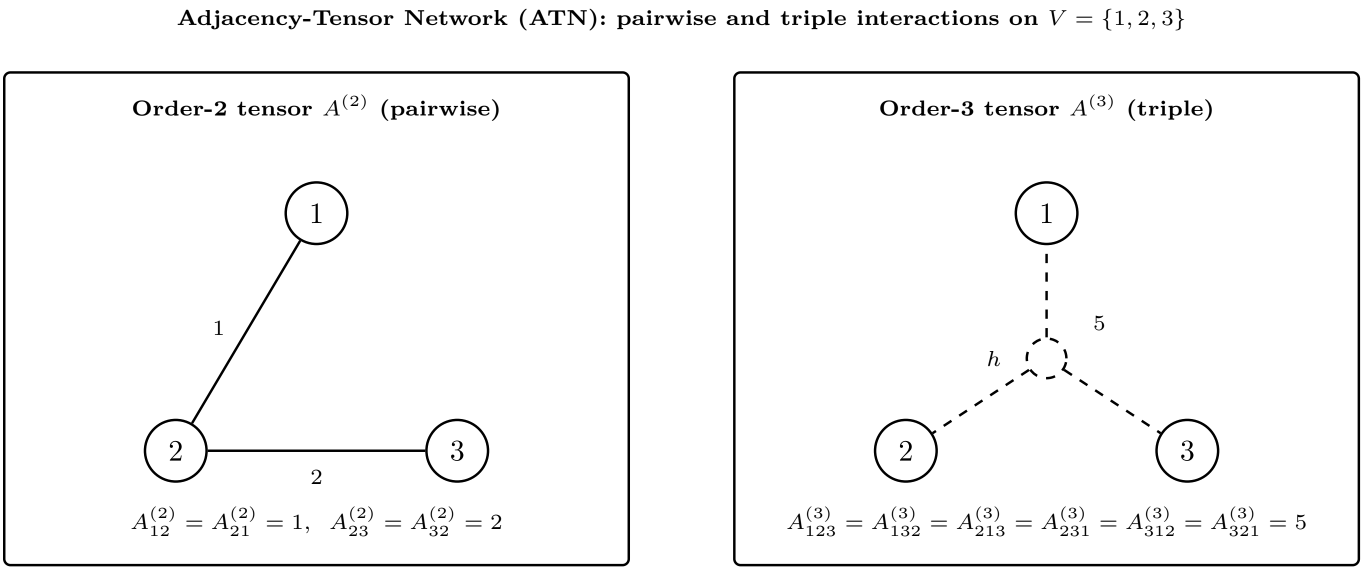


Figure 4.4: An ATN with pairwise and triple interactions on $V = \{1, 2, 3\}$. The left panel visualizes nonzero entries of $A^{(2)}$, and the right panel visualizes the nonzero symmetric triple interaction in $A^{(3)}$.

as follows.

Pairwise tensor. Let $A^{(2)} \in \mathbb{R}^{3\times 3}$ be symmetric with nonzero entries

$$A^{(2)}_{12} = A^{(2)}_{21} = 1, \qquad A^{(2)}_{23} = A^{(2)}_{32} = 2,$$

and all other entries 0. Thus, the pairwise interaction $\{1, 2\}$ has weight 1, and $\{2, 3\}$ has weight 2.

Triple tensor. Let $A^{(3)} \in \mathbb{R}^{3\times 3\times 3}$ be fully symmetric with

$$A^{(3)}_{123} = A^{(3)}_{132} = A^{(3)}_{213} = A^{(3)}_{231} = A^{(3)}_{312} = A^{(3)}_{321} = 5,$$

and all other entries 0 (in particular, entries with repeated indices are 0). This encodes a 3-way interaction among $\{1, 2, 3\}$ of weight 5.

Then $\mathbf{A}$ is an Adjacency-Tensor Network in the sense of Definition 4.8.1, where nonzero entries of $A^{(2)}$ and $A^{(3)}$ specify weighted 2-way and 3-way interactions, respectively. An overview diagram of this example is provided in Fig. 4.4.

Theorem 4.8.3 (Well-definedness of Adjacency-Tensor Networks)**.** *Let $V = \{1, \dots, n\}$ be a finite vertex set with $n \geq 1$, and fix an integer $K \geq 2$. For each $k \in \{2, \dots, K\}$, define*

$$\mathcal{T}_k(V) := \mathbb{R}^{V^k} = \{\, \alpha : V^k \to \mathbb{R} \,\}.$$

Then the following hold.

1. *For each k, there is a canonical identification*

$$\mathcal{T}_k(V) \;\cong\; \mathbb{R}^{n\times\cdots\times n} \quad (k \text{ factors}),$$

given by

$$\alpha \longmapsto A^{(k)}, \qquad A^{(k)}_{i_1\cdots i_k} := \alpha(i_1, \dots, i_k) \quad ((i_1, \dots, i_k) \in V^k).$$

Hence a k-way adjacency tensor is a well-defined object.

2. *The family space*

$$\mathcal{T}_{\leq K}(V) := \prod_{k=2}^{K} \mathcal{T}_k(V)$$

is well-defined (a finite Cartesian product of sets, in fact a finite-dimensional real vector space). Each element

$$\mathbf{A} = (A^{(2)}, A^{(3)}, \dots, A^{(K)}) \in \mathcal{T}_{\leq K}(V)$$

therefore defines a well-formed Adjacency-Tensor Network (ATN) on V.

3. *For each k, the symmetry condition in the undirected k-uniform case,*

$$A^{(k)}_{i_{\pi(1)}\cdots i_{\pi(k)}} = A^{(k)}_{i_1\cdots i_k} \qquad (\forall (i_1,\dots,i_k) \in V^k,\ \forall \pi \in S_k),$$

is well-defined and is equivalent to saying that the associated function $\alpha_k : V^k \to \mathbb{R}$ is constant on the S_k-orbits of V^k under coordinate permutation.

Consequently, Definition 4.8.1 is well-defined.

Proof. For each $k \in \{2,\dots,K\}$, the set V^k is finite because V is finite. Hence $\mathcal{T}_k(V) = \mathbb{R}^{V^k}$ is a well-defined set of real-valued functions on V^k.

(1) Canonical tensor indexing. Since $V = \{1,\dots,n\}$, every element of V^k is a k-tuple $(i_1,\dots,i_k)$ with $i_j \in \{1,\dots,n\}$. Thus each $\alpha \in \mathcal{T}_k(V)$ determines a unique array

$$A^{(k)} = (A^{(k)}_{i_1\cdots i_k})_{(i_1,\dots,i_k)\in V^k}$$

by the rule $A^{(k)}_{i_1\cdots i_k} := \alpha(i_1,\dots,i_k)$. Conversely, any k-indexed real array $(A^{(k)}_{i_1\cdots i_k})$ defines a unique function

$$\alpha : V^k \to \mathbb{R}, \qquad \alpha(i_1,\dots,i_k) = A^{(k)}_{i_1\cdots i_k}.$$

These constructions are inverse to each other, so

$$\mathcal{T}_k(V) \cong \mathbb{R}^{n\times\cdots\times n} \quad (k \text{ factors}).$$

Hence the notation $A^{(k)} \in \mathbb{R}^{n\times\cdots\times n}$ is mathematically well-defined.

(2) Family of tensors. Because the index set $\{2,\dots,K\}$ is finite, the Cartesian product

$$\mathcal{T}_{\le K}(V) = \prod_{k=2}^{K} \mathcal{T}_k(V)$$

is well-defined. An element $\mathbf{A} \in \mathcal{T}_{\le K}(V)$ is exactly a family

$$\mathbf{A} = (A^{(2)}, A^{(3)}, \dots, A^{(K)}), \qquad A^{(k)} \in \mathcal{T}_k(V) \cong \mathbb{R}^{n\times\cdots\times n},$$

which is precisely the data specified in Definition 4.8.1. Therefore an ATN is a well-formed mathematical object.

(3) Symmetry condition. Fix k. For any permutation $\pi \in S_k$, the tuple $(i_{\pi(1)},\dots,i_{\pi(k)})$ belongs to V^k whenever $(i_1,\dots,i_k) \in V^k$. Hence the expression

$$A^{(k)}_{i_{\pi(1)}\cdots i_{\pi(k)}}$$

is defined, and so the equality

$$A^{(k)}_{i_{\pi(1)}\cdots i_{\pi(k)}} = A^{(k)}_{i_1\cdots i_k}$$

is a well-posed condition.

Let $\alpha_k : V^k \to \mathbb{R}$ be the function corresponding to $A^{(k)}$. Then the above equality is equivalent to

$$\alpha_k(i_{\pi(1)},\dots,i_{\pi(k)}) = \alpha_k(i_1,\dots,i_k) \quad (\forall (i_1,\dots,i_k) \in V^k,\ \forall \pi \in S_k),$$

which exactly means that α_k is constant on each orbit of the natural S_k-action on V^k by coordinate permutation. Thus the undirected symmetry requirement is also well-defined.

Combining (1)–(3), Definition 4.8.1 is well-defined. □

5 Semantic, Compositional, Knowledge, and Logical Family

Semantic, compositional, knowledge, and logical families model higher-order systems through meaning, interfaces, composition, inference, and structured knowledge, emphasizing how relations, transformations, and interpretations interact across complex hierarchical networked representations. For reference, the semantic, compositional, knowledge, and logical higher-order structures treated in this book are listed in Table 5.1.

Table 5.1: Semantic, compositional, knowledge, and logical higher-order structures treated in this book.

Concept	Concise description
Open Hypergraph	A hypergraph with designated input and output interfaces, supporting compositional modeling of open higher-order interactions.
Heterogeneous Graph, HyperGraph, and SuperHyperGraph	Typed graph-based structures whose vertices and edges may belong to different categories or semantic classes.
Knowledge Graph, HyperGraph, and SuperHyperGraph	Semantic structures encoding entities and relations, extended from binary facts to higher-arity and hierarchical knowledge representations.
Petri Net	A bipartite semantic model of places and transitions, representing concurrency, synchronization, resource flow, and reachability.
Port Graph	A graph with explicit ports attached to nodes, enabling structured interfaces and fine-grained connection semantics.
Port HyperGraph and Port SuperHyperGraph	Port-based higher-order structures in which hyperedges or superhyperedges connect ports rather than directly connecting nodes.
Open Hypergraph and Open SuperHyperGraph	Open higher-order structures with boundary interfaces, allowing compositional connection of hypergraphs and superhypergraphs.
Combinatorial Map	A discrete topological-combinatorial structure encoding adjacency, incidence, and embedding information through darts and permutations.
Cognitive HyperGraphs and Cognitive SuperHyperGraphs	Hypergraph-based cognitive models representing concepts, associations, and higher-level grouped cognitive structures.
Multimodal Graph, HyperGraph, and SuperHyperGraph	Structures combining multiple modalities or interaction channels on a common graph, hypergraph, or superhypergraph framework.
Operadic Interaction Graph (OIG)	A compositional graph model based on operadic ideas, capturing structured composition of multi-input interactions.
Symmetric Monoidal Wiring Graph (SMWG)	A wiring-style higher-order model expressing compositional systems through symmetric monoidal connections and interfaces.
Relational-Arity Graph (RAG)	A graph framework organizing relations according to arity, bridging ordinary graphs and higher-arity relational structures.
Closure-Implication Graph (CIG)	A logical graph model encoding implication and closure behavior among elements, attributes, or derived statements.

Continued on the next page

Table 5.1 (continued)

Concept	Concise description
Coalgebraic Nested-Neighborhood Graph (CNNG)	A coalgebra-inspired graph structure representing recursively nested neighborhood semantics and hierarchical observation patterns.
Curried Graph	A graph whose vertices are curried functions and whose edges preserve typed functional compatibility through commuting conditions.
Depth-r iterated subdivisions of polyhedral complexes	Repeated subdivision structures on polyhedral complexes, recording multilevel refinement of combinatorial-geometric organization.
Sheaf HyperGraph / Sheaf SuperHyperGraph	Hypergraph and superhypergraph structures equipped with sheaf data, assigning local spaces and compatible restriction maps.
Fibered HyperGraph / Fibered SuperHyperGraph	Hypergraph and superhypergraph models enriched with fibers over vertices and edgewise relations among local states.
Galois HyperGraph / Galois SuperHyperGraph	Formal-context-based higher-order structures whose hyperedges arise from Galois-closed families of vertices or supervertices.
Rewrite HyperGraph / Rewrite SuperHyperGraph	Hypergraph and superhypergraph frameworks equipped with rewrite rules for rule-based transformation of higher-order structures.
Uncertain SuperHyperGraph	A superhypergraph framework incorporating uncertainty in vertices, edges, or incidence through set-valued or uncertainty-aware semantics.
Functorial SuperHyperGraph	A categorical superhypergraph model emphasizing morphisms and functorial behavior between hierarchical higher-order structures.
Topological SuperHyperGraph	A topological superhypergraph model combining superhypergraph nesting with topological structure, enabling continuous, spatial, and inclusion-based representations.
Motif Hypergraphs and Motif SuperHypergraphs	Higher-order motif-based structures that encode recurring subgraph patterns either as hyperedges on supporting vertices or as supervertices linked by higher-order relations.
Molecular SuperHyperGraphs	Hierarchical chemical higher-order structures organizing atoms, bonds, fragments, and larger molecular units across iterated powerset levels.

5.1 Heterogeneous Graph, HyperGraph, and SuperHyperGraph

A heterogeneous graph is a graph in which vertices and edges may belong to different types, allowing multiple classes of objects and relations to be represented within a single structure [232, 233, 234]. A heterogeneous hypergraph is a hypergraph in which vertices and hyperedges may have different types, enabling the modeling of typed multiway relations among diverse objects in one framework [235, 236, 237]. A heterogeneous superhypergraph is a superhypergraph with typed higher-order vertices and superhyperedges, designed to represent heterogeneous hierarchical relations through iterated set-based constructions.

Definition 5.1.1 (Heterogeneous Graph)**.** [232, 233, 234] Let T_V and T_E be nonempty sets, called the sets of vertex-types and edge-types, respectively. A *heterogeneous graph* is a sextuple

$$G = (V, E, T_V, T_E, \tau_V, \tau_E),$$

where

1. V is a finite nonempty set of vertices;

2. $$E \subseteq \{\{u, v\} \subseteq V : u \neq v\}$$

is the set of edges;

3. $\tau_V : V \to T_V$ is a vertex-type map;

4. $\tau_E : E \to T_E$ is an edge-type map.

If either $|T_V| > 1$ or $|T_E| > 1$, then G is called *heterogeneous*.

Definition 5.1.2 (Heterogeneous HyperGraph)**.** Let T_V and T_E be nonempty sets. A *heterogeneous hypergraph* is a sextuple

$$H = (V, E, T_V, T_E, \tau_V, \tau_E),$$

where

1. V is a finite nonempty set of vertices;

2. $$E \subseteq \mathcal{P}^*(V) := \mathcal{P}(V) \setminus \{\emptyset\}$$

   is a finite set of nonempty subsets of V, called hyperedges;

3. $\tau_V : V \to T_V$ is a vertex-type map;

4. $\tau_E : E \to T_E$ is a hyperedge-type map.

If either $|T_V| > 1$ or $|T_E| > 1$, then H is called *heterogeneous*.

Definition 5.1.3 (Heterogeneous n-SuperHyperGraph)**.** Let V_0 be a finite nonempty base set, and let $n \in \mathbb{N} \cup \{0\}$. Define the iterated powersets recursively by

$$\mathcal{P}^0(V_0) := V_0, \qquad \mathcal{P}^{k+1}(V_0) := \mathcal{P}\big(\mathcal{P}^k(V_0)\big) \quad (k \geq 0).$$

Let T_V and T_E be nonempty sets. A *heterogeneous n-SuperHyperGraph* is a sextuple

$$\mathcal{H}^{(n)} = (V, E, T_V, T_E, \tau_V, \tau_E),$$

where

1. $$V \subseteq \mathcal{P}^n(V_0)$$

   is a finite set, whose elements are called n-supervertices;

2. $$E \subseteq \mathcal{P}^*(V)$$

   is a finite set of nonempty subsets of V, whose elements are called n-superhyperedges;

3. $\tau_V : V \to T_V$ is a supervertex-type map;

4. $\tau_E : E \to T_E$ is a superhyperedge-type map.

If either $|T_V| > 1$ or $|T_E| > 1$, then $\mathcal{H}^{(n)}$ is called *heterogeneous*.

Example 5.1.4 (A concrete Heterogeneous 1-SuperHyperGraph)**.** Let the finite base set be

$$V_0 = \{\mathsf{Alice}, \mathsf{Bob}, \mathsf{Carol}, \mathsf{DataA}, \mathsf{DataB}, \mathsf{Server}\},$$

and take $n = 1$. Then

$$\mathcal{P}^1(V_0) = \mathcal{P}(V_0),$$

so each 1-supervertex is a subset of V_0.

Define the set of 1-supervertices by

$$V = \{v_1, v_2, v_3, v_4\} \subseteq \mathcal{P}(V_0),$$

where

$$v_1 = \{\mathsf{Alice}, \mathsf{Bob}\}, \qquad v_2 = \{\mathsf{Carol}\},$$

$$v_3 = \{\mathsf{DataA}, \mathsf{DataB}\}, \qquad v_4 = \{\mathsf{Bob}, \mathsf{Server}\}.$$

Next, define the supervertex-type set by

$$T_V = \{\mathsf{Team}, \mathsf{Individual}, \mathsf{DatasetGroup}, \mathsf{TechnicalUnit}\},$$

and define the type map

$$\tau_V : V \to T_V$$

by

$$\tau_V(v_1) = \mathsf{Team}, \qquad \tau_V(v_2) = \mathsf{Individual},$$

$$\tau_V(v_3) = \mathsf{DatasetGroup}, \qquad \tau_V(v_4) = \mathsf{TechnicalUnit}.$$

Now define the set of 1-superhyperedges by

$$E = \{e_1, e_2, e_3\} \subseteq \mathcal{P}^*(V),$$

where

$$e_1 = \{v_1, v_2\}, \qquad e_2 = \{v_1, v_3, v_4\}, \qquad e_3 = \{v_2, v_4\}.$$

Clearly each e_i is a nonempty subset of V, so indeed $E \subseteq \mathcal{P}^*(V)$.

Define the superhyperedge-type set by

$$T_E = \{\mathsf{Coordination}, \mathsf{AccessControl}, \mathsf{Maintenance}\},$$

and define the edge-type map

$$\tau_E : E \to T_E$$

by

$$\tau_E(e_1) = \mathsf{Coordination}, \qquad \tau_E(e_2) = \mathsf{AccessControl}, \qquad \tau_E(e_3) = \mathsf{Maintenance}.$$

Therefore,

$$\mathcal{H}^{(1)} = (V, E, T_V, T_E, \tau_V, \tau_E)$$

is a Heterogeneous 1-SuperHyperGraph.

The supervertex $v_1 = \{\mathsf{Alice}, \mathsf{Bob}\}$ is a team-type unit, $v_2 = \{\mathsf{Carol}\}$ is an individual-type unit, $v_3 = \{\mathsf{DataA}, \mathsf{DataB}\}$ is a dataset-group unit, and $v_4 = \{\mathsf{Bob}, \mathsf{Server}\}$ is a technical unit. The hyperedge e_1 represents a coordination relation between a team and an individual, e_2 represents an access-control relation among a team, a dataset-group, and a technical unit, and e_3 represents a maintenance relation between an individual and a technical unit.

Since

$$|T_V| = 4 > 1 \qquad \text{and} \qquad |T_E| = 3 > 1,$$

the structure is heterogeneous in both its supervertex types and superhyperedge types. An illustration is given in Fig. 5.1.

5.2 Knowledge Graph, HyperGraph, and SuperHyperGraph

A knowledge graph represents entities and their binary relations as structured facts, enabling semantic reasoning, retrieval, and integration across domains [238, 239, 240]. A knowledge hypergraph models entities through higher arity relations, capturing multi-entity facts and richer semantic structures within unified knowledge bases [241, 242]. A knowledge superhypergraph represents hierarchical set-based entities and relations, capturing higher-order semantic facts across multiple abstraction levels within frameworks coherently.

Definition 5.2.1 (Knowledge Graph)**.** [238, 239, 240] Let E be a finite set of entities and let R be a finite set of binary relations. Define

$$\tau := \{\, r(e_1, e_2) \mid r \in R,\ e_1, e_2 \in E \,\}.$$

A *knowledge graph* is a triple

$$KG = (E, R, \tau_0),$$

where $\tau_0 \subseteq \tau$ is the set of true binary facts.

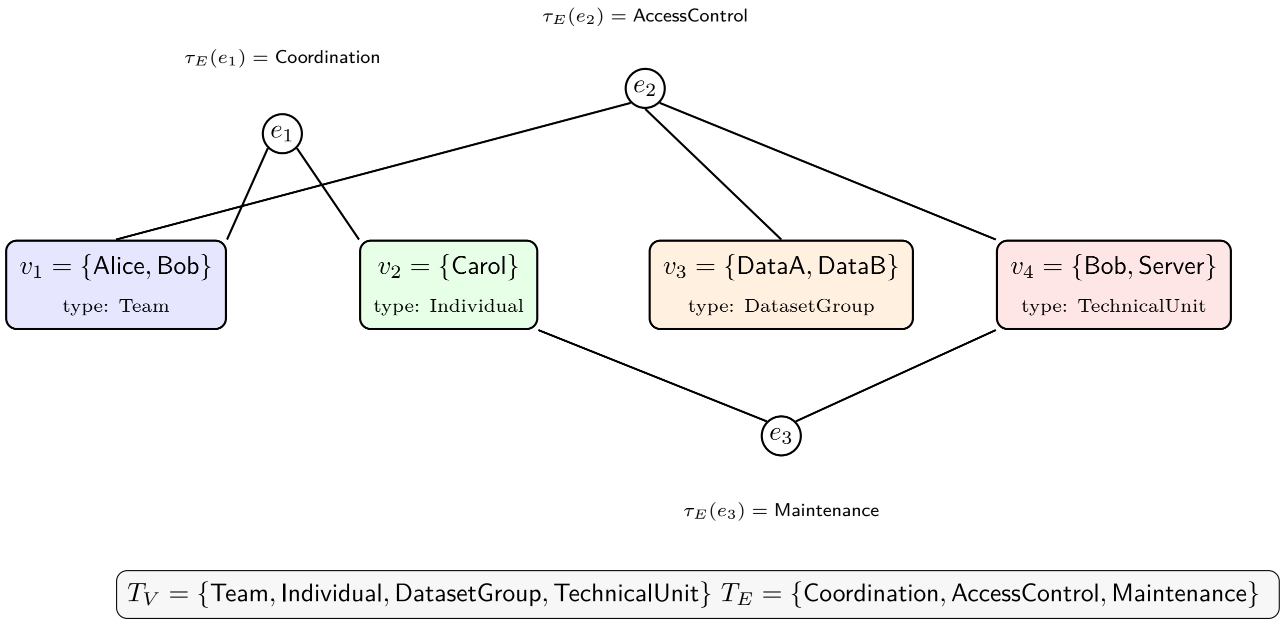


Figure 5.1: A concrete Heterogeneous 1-SuperHyperGraph. The supervertices are typed subsets of the base set, and the superhyperedges are nonempty subsets of the common supervertex set equipped with edge-types.

Definition 5.2.2 (Knowledge HyperGraph)**.** [243, 244] Let E be a finite set of entities and let R be a finite set of relations equipped with an arity map

$$\mathrm{ar} : R \to \mathbb{N}.$$

Define

$$\tau := \{\, r(e_1, \ldots, e_{\mathrm{ar}(r)}) \mid r \in R,\ e_i \in E \text{ for all } i \,\}.$$

A *knowledge hypergraph* is a triple

$$KH = (E, R, \tau_0),$$

where $\tau_0 \subseteq \tau$ is the set of true facts.

Equivalently, each true fact

$$r(e_1, \ldots, e_{\mathrm{ar}(r)}) \in \tau_0$$

may be viewed as a hyperedge joining the participating entities.

Definition 5.2.3 (Knowledge n-SuperHyperGraph)**.** [245, 246] Let E_0 be a finite set of base entities and let R_0 be a finite set of base relations. For each integer $k \geq 0$, define iterated powersets by

$$\mathcal{P}^0(E_0) := E_0, \qquad \mathcal{P}^{k+1}(E_0) := \mathcal{P}\big(\mathcal{P}^k(E_0)\big),$$

and similarly

$$\mathcal{P}^0(R_0) := R_0, \qquad \mathcal{P}^{k+1}(R_0) := \mathcal{P}\big(\mathcal{P}^k(R_0)\big).$$

Fix $n \geq 0$. Let

$$V \subseteq \mathcal{P}^n(E_0)$$

be a finite set, whose elements are called *n-superentities*, and let

$$\mathcal{R} \subseteq \mathcal{P}^n(R_0)$$

be a finite set, whose elements are called *n-superrelations*. Assume that $\mathcal{R}$ is equipped with an arity map

$$\mathrm{ar}^{(n)} : \mathcal{R} \to \mathbb{N}.$$

Define

$$\tau^{(n)} := \left\{ r(v_1, \ldots, v_{\mathrm{ar}^{(n)}(r)}) \;\middle|\; r \in \mathcal{R},\ v_i \in V \right\}.$$

A *knowledge n-SuperHyperGraph* is a triple

$$KH^{(n)} = (V, \mathcal{R}, \tau_0^{(n)}),$$

where

$$\tau_0^{(n)} \subseteq \tau^{(n)}$$

is the set of true n-superfacts.

Example 5.2.4 (A concrete Knowledge 1-SuperHyperGraph)**.** Let the base entity set be

$$E_0 = \{\mathsf{Alice}, \mathsf{Bob}, \mathsf{Carol}, \mathsf{ProjectX}, \mathsf{ProjectY}\},$$

and let the base relation set be

$$R_0 = \{\mathsf{worksOn}, \mathsf{collaboratesWith}, \mathsf{supports}\}.$$

Take $n = 1$. Then

$$\mathcal{P}^1(E_0) = \mathcal{P}(E_0), \qquad \mathcal{P}^1(R_0) = \mathcal{P}(R_0).$$

Define the set of 1-superentities by

$$V = \{v_1, v_2, v_3\} \subseteq \mathcal{P}(E_0),$$

where

$$v_1 = \{\mathsf{Alice}, \mathsf{Bob}\}, \qquad v_2 = \{\mathsf{Carol}\}, \qquad v_3 = \{\mathsf{ProjectX}, \mathsf{ProjectY}\}.$$

Thus v_1 is a team of two researchers, v_2 is a singleton researcher-group, and v_3 is a project-group.
Next define the set of 1-superrelations by

$$\mathcal{R} = \{r_1, r_2\} \subseteq \mathcal{P}(R_0),$$

where

$$r_1 = \{\mathsf{collaboratesWith}, \mathsf{worksOn}\}, \qquad r_2 = \{\mathsf{supports}\}.$$

Equip $\mathcal{R}$ with the arity map

$$\mathrm{ar}^{(1)}(r_1) = 2, \qquad \mathrm{ar}^{(1)}(r_2) = 2.$$

Hence the set of all admissible 1-superfacts is

$$\tau^{(1)} = \{\, r(v_i, v_j) \mid r \in \mathcal{R},\ v_i, v_j \in V \,\}.$$

Now choose the following subset of true 1-superfacts:

$$\tau_0^{(1)} = \{r_1(v_1, v_2),\ r_1(v_1, v_3),\ r_2(v_3, v_1)\}.$$

Therefore

$$KH^{(1)} = (V, \mathcal{R}, \tau_0^{(1)})$$

is a Knowledge 1-SuperHyperGraph in the sense of the above definition.
The true superfact

$$r_1(v_1, v_2)$$

means that the grouped relation

$$\{\mathsf{collaboratesWith}, \mathsf{worksOn}\}$$

holds from the superentity $v_1 = \{\mathsf{Alice}, \mathsf{Bob}\}$ to the superentity $v_2 = \{\mathsf{Carol}\}$. Likewise,

$$r_1(v_1, v_3)$$

states that the researcher-group v_1 is connected to the project-group v_3 by the same grouped semantic relation, and

$$r_2(v_3, v_1)$$

states that the project-group v_3 supports the researcher-group v_1.
Thus this example illustrates how a knowledge structure may be lifted from base entities and base relations to grouped entities and grouped relations through the powerset construction.

5.3 Petri Net

A Petri net is a directed bipartite graph of places and transitions, modeling concurrency, synchronization, resource flow, and reachability dynamically [247, 248, 249]. Related concepts such as the Fuzzy Petri Net [250, 251] and the Neutrosophic Petri Net [252, 253] are also known.

Definition 5.3.1 (Petri net)**.** [247, 248, 249] A *Petri net* is a tuple

$$\mathcal{N} = (P, T, F, W),$$

where

1. P is a finite set of *places*;
2. T is a finite set of *transitions*;
3. $P \cap T = \varnothing$;
4. $$F \subseteq (P \times T) \cup (T \times P)$$
   is the *flow relation* (or set of directed arcs);
5. $$W : F \to \mathbb{N}_{>0}$$
   is the *weight function*.

Thus (P, T, F) is a directed bipartite graph: arcs are allowed only from places to transitions or from transitions to places.

Definition 5.3.2 (Preset and postset)**.** Let $\mathcal{N} = (P, T, F, W)$ be a Petri net. For each transition $t \in T$, define its *preset* and *postset* by

$$^{\bullet}t := \{\, p \in P \mid (p, t) \in F \,\}, \qquad t^{\bullet} := \{\, p \in P \mid (t, p) \in F \,\}.$$

Similarly, for each place $p \in P$, define

$$^{\bullet}p := \{\, t \in T \mid (t, p) \in F \,\}, \qquad p^{\bullet} := \{\, t \in T \mid (p, t) \in F \,\}.$$

Definition 5.3.3 (Marking and marked Petri net)**.** A *marking* of a Petri net $\mathcal{N} = (P, T, F, W)$ is a function

$$M : P \to \mathbb{N}_0.$$

For each place $p \in P$, the value $M(p)$ is called the number of *tokens* in p.

A *marked Petri net* is a pair

$$(\mathcal{N}, M_0),$$

where $\mathcal{N}$ is a Petri net and M_0 is an initial marking.

Definition 5.3.4 (Enabled transition and firing rule)**.** Let $(\mathcal{N}, M)$ be a marked Petri net, where

$$\mathcal{N} = (P, T, F, W).$$

A transition $t \in T$ is said to be *enabled* at the marking M if

$$M(p) \geq W(p, t) \qquad \text{for all } p \in {}^{\bullet}t.$$

If t is enabled, then t may *fire*, producing a new marking

$$M' : P \to \mathbb{N}_0$$

defined by

$$M'(p) = \begin{cases} M(p) - W(p,t) + W(t,p), & \text{if } p \in {}^{\bullet}t \cap t^{\bullet}, \\ M(p) - W(p,t), & \text{if } p \in {}^{\bullet}t \setminus t^{\bullet}, \\ M(p) + W(t,p), & \text{if } p \in t^{\bullet} \setminus {}^{\bullet}t, \\ M(p), & \text{if } p \notin {}^{\bullet}t \cup t^{\bullet}, \end{cases}$$

where, by convention, $W(x, y) = 0$ whenever $(x, y) \notin F$. Equivalently, for every $p \in P$,

$$M'(p) = M(p) - W(p, t) + W(t, p).$$

Example 5.3.5 (A simple document-processing Petri net)**.** Consider a system in which one document is first prepared and then approved.

Let

$$P = \{p_1, p_2, p_3\}, \qquad T = \{t_1, t_2\},$$

where:

- p_1 represents *document available for processing*,
- p_2 represents *document under review*,
- p_3 represents *document approved*,
- t_1 represents *start review*,
- t_2 represents *approve document*.

Define the flow relation

$$F = \{(p_1, t_1), (t_1, p_2), (p_2, t_2), (t_2, p_3)\},$$

and let all arc weights be equal to 1, i.e.

$$W(f) = 1 \qquad \text{for all } f \in F.$$

Take the initial marking

$$M_0(p_1) = 1, \qquad M_0(p_2) = 0, \qquad M_0(p_3) = 0.$$

Thus the initial state contains one token in p_1, meaning that one document is ready to be reviewed.

At the marking M_0, the transition t_1 is enabled because

$$M_0(p_1) = 1 \geq W(p_1, t_1) = 1.$$

After firing t_1, the new marking M_1 is

$$M_1(p_1) = 0, \qquad M_1(p_2) = 1, \qquad M_1(p_3) = 0.$$

Now t_2 is enabled, and after firing t_2, one obtains

$$M_2(p_1) = 0, \qquad M_2(p_2) = 0, \qquad M_2(p_3) = 1.$$

Hence the token has moved from p_1 to p_3 through the intermediate review state p_2.

5.4 Port Graph

A port graph equips nodes with explicit connection ports, enabling structured interaction patterns, fine-grained interfaces, and rule-based rewriting [254, 255, 256].

Definition 5.4.1 (Port Graph)**.** [257] A *port graph* is a tuple

$$G = (V, P, E, \mathrm{Attach}, \mathrm{Connect}),$$

where

1. V is a finite set of *nodes*;
2. P is a finite set of *ports*;
3. E is a finite set of *edges*;
4. $$\mathrm{Attach} : P \to V$$
   is the *attachment map*, assigning to each port the node to which it belongs;
5. $$\mathrm{Connect} : E \to \big\{\{p, q\} \subseteq P \mid p \neq q\big\}$$
   is the *connection map*, assigning to each edge an unordered pair of distinct ports.

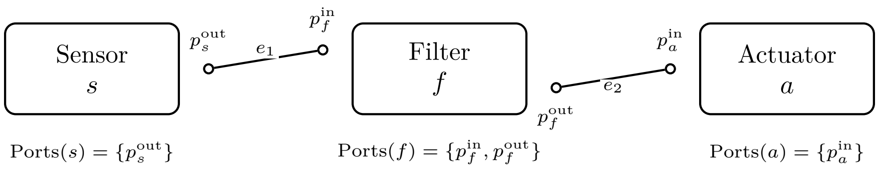


Figure 5.2: A simple port graph. Edges connect ports rather than directly connecting nodes.

Remark 5.4.2. *For a node $v \in V$, its* interface *is the set*

$$\text{Ports}(v) := \text{Attach}^{-1}(\{v\}) \subseteq P.$$

Thus, a port graph differs from an ordinary graph in that adjacency is mediated by ports. Two nodes may be connected through specific designated ports, which allows one to encode structured interfaces and fine-grained interaction patterns.

Example 5.4.3 (A simple signal-processing port graph)**.** Consider a small signal-processing pipeline with three components: a sensor, a filter, and an actuator.

Let the node set be

$$V = \{s, f, a\},$$

where

$$s = \text{Sensor}, \qquad f = \text{Filter}, \qquad a = \text{Actuator}.$$

Define the port set

$$P = \{p_s^{\text{out}},\, p_f^{\text{in}},\, p_f^{\text{out}},\, p_a^{\text{in}}\}.$$

The attachment map $\text{Attach} : P \to V$ is given by

$$\text{Attach}(p_s^{\text{out}}) = s, \qquad \text{Attach}(p_f^{\text{in}}) = f, \qquad \text{Attach}(p_f^{\text{out}}) = f, \qquad \text{Attach}(p_a^{\text{in}}) = a.$$

Thus the interfaces are

$$\text{Ports}(s) = \{p_s^{\text{out}}\}, \qquad \text{Ports}(f) = \{p_f^{\text{in}}, p_f^{\text{out}}\}, \qquad \text{Ports}(a) = \{p_a^{\text{in}}\}.$$

Let the edge set be

$$E = \{e_1, e_2\},$$

and define the connection map by

$$\text{Connect}(e_1) = \{p_s^{\text{out}}, p_f^{\text{in}}\}, \qquad \text{Connect}(e_2) = \{p_f^{\text{out}}, p_a^{\text{in}}\}.$$

Hence

$$G = (V, P, E, \text{Attach}, \text{Connect})$$

is a port graph in the sense of Definition 5.4.1.

Intuitively, the first edge e_1 sends the sensor output to the filter input, and the second edge e_2 sends the filter output to the actuator input. An illustration is given in Fig. 5.2.

5.5 Port HyperGraph and Port SuperHyperGraph

In this section, we extend the notion of a port graph to hypergraphs and superhypergraphs. The main idea is that hyperedges (or superhyperedges) are incident not directly with vertices, but with *ports*, and each port is attached to exactly one vertex or supervertex.

Definition 5.5.1 (Port HyperGraph)**.** A *Port HyperGraph* is a quintuple

$$\mathcal{H}^{\text{port}} = (V, \Pi, E, \alpha, \iota),$$

where

1. V is a finite nonempty set of *vertices*;
2. Π is a finite set of *ports*;

3. E is a finite set of *hyperedge identifiers*;

4. $$\alpha : \Pi \to V$$

   is the *port-attachment map*, assigning to each port the unique vertex to which it belongs;

5. $$\iota : E \to \mathcal{P}^*(\Pi)$$

   is the *port-incidence map*, assigning to each hyperedge identifier a nonempty set of ports.

Remark 5.5.2. *For each vertex $v \in V$, its* port interface *is the set*

$$\mathrm{Port}(v) := \alpha^{-1}(\{v\}) \subseteq \Pi.$$

Thus a Port HyperGraph differs from an ordinary hypergraph in that a hyperedge is incident with a set of ports, rather than directly with a set of vertices.

Remark 5.5.3. *If every hyperedge $e \in E$ satisfies*

$$|\iota(e)| = 2,$$

then $\mathcal{H}^{\mathrm{port}}$ reduces to a graph-like port structure in which each edge joins exactly two ports. Hence Port Graphs may be viewed as a special binary case of Port HyperGraphs.

Theorem 5.5.4 (Well-definedness of Port HyperGraphs)**.** *Let V be a finite nonempty set, let Π and E be finite sets, and let*

$$\alpha : \Pi \to V, \qquad \iota : E \to \mathcal{P}^*(\Pi)$$

be well-defined maps. Then

$$\mathcal{H}^{\mathrm{port}} = (V, \Pi, E, \alpha, \iota)$$

is a well-defined Port HyperGraph.

Proof. Since V is a finite nonempty set and Π is a finite set, the map

$$\alpha : \Pi \to V$$

is a legitimate set-theoretic assignment attaching each port to a unique vertex.

Since Π is a set, its nonempty powerset

$$\mathcal{P}^*(\Pi) = \mathcal{P}(\Pi) \setminus \{\varnothing\}$$

is well-defined. Therefore the map

$$\iota : E \to \mathcal{P}^*(\Pi)$$

assigns to each hyperedge identifier $e \in E$ a nonempty subset of ports.

Hence all components appearing in Definition 5.5.1 are well-defined and mutually compatible. Therefore

$$\mathcal{H}^{\mathrm{port}} = (V, \Pi, E, \alpha, \iota)$$

is a well-defined Port HyperGraph. □

Definition 5.5.5 (Port n-SuperHyperGraph)**.** Let V_0 be a finite nonempty base set, and let $n \in \mathbb{N}_0$. A *Port n-SuperHyperGraph* is a quintuple

$$\mathcal{SH}^{(n),\mathrm{port}} = (V, \Pi, E, \alpha, \iota),$$

where

1. $$V \subseteq \mathcal{P}^n(V_0)$$

   is a finite set of *n-supervertices*;

2. Π is a finite set of *ports*;

3. E is a finite set of *superhyperedge identifiers*;

4.
$$\alpha : \Pi \to V$$
is the *port-attachment map*, assigning each port to a unique n-supervertex;

5.
$$\iota : E \to \mathcal{P}^*(\Pi)$$
is the *port-superincidence map*, assigning to each superhyperedge identifier a nonempty set of ports.

Remark 5.5.6. *For each supervertex $v \in V$, its* port interface *is*

$$\mathrm{Port}(v) := \alpha^{-1}(\{v\}) \subseteq \Pi.$$

Thus a Port n-SuperHyperGraph is a superhypergraph-like structure in which incidence is mediated by ports attached to n-supervertices.

Remark 5.5.7. *A* Port SuperHyperGraph *means a Port n-SuperHyperGraph for some fixed*

$$n \in \mathbb{N}_0.$$

In particular, when $n = 0$, one has
$$\mathcal{P}^0(V_0) = V_0,$$
so a Port 0-SuperHyperGraph is exactly a Port HyperGraph.

Theorem 5.5.8 (Well-definedness of Port n-SuperHyperGraphs)**.** *Let V_0 be a finite nonempty set, let $n \in \mathbb{N}_0$, let*
$$V \subseteq \mathcal{P}^n(V_0)$$
be a finite set, let Π and E be finite sets, and let

$$\alpha : \Pi \to V, \qquad \iota : E \to \mathcal{P}^*(\Pi)$$

be well-defined maps. Then
$$\mathcal{SH}^{(n),\mathrm{port}} = (V, \Pi, E, \alpha, \iota)$$
is a well-defined Port n-SuperHyperGraph.

Proof. First, since V_0 is a finite nonempty set and $n \in \mathbb{N}_0$, the iterated powerset

$$\mathcal{P}^n(V_0)$$

is well-defined by finite recursion. Hence the condition

$$V \subseteq \mathcal{P}^n(V_0)$$

is meaningful, and V is a well-defined finite set of n-supervertices.

Next, since Π is a finite set, its nonempty powerset

$$\mathcal{P}^*(\Pi) = \mathcal{P}(\Pi) \setminus \{\varnothing\}$$

is well-defined. Therefore the map
$$\iota : E \to \mathcal{P}^*(\Pi)$$
assigns to each superhyperedge identifier a nonempty set of ports.

Moreover, the map
$$\alpha : \Pi \to V$$
is well-defined, so each port is attached to a unique n-supervertex.

Thus all components listed in Definition 5.5.5 are well-defined and mutually compatible. Therefore

$$\mathcal{SH}^{(n),\mathrm{port}} = (V, \Pi, E, \alpha, \iota)$$

is a well-defined Port n-SuperHyperGraph. □

Corollary 5.5.9. *Every Port HyperGraph is a Port 0-SuperHyperGraph.*

Proof. If $n = 0$, then
$$\mathcal{P}^0(V_0) = V_0,$$
so the supervertex set is an ordinary vertex set. Hence Definition 5.5.5 reduces exactly to Definition 5.5.1. □

5.6 Open Hypergraph and Open SuperHyperGraph

An open hypergraph is a hypergraph with designated inputs and outputs, supporting modular composition of networked systems through cospans categorically.

Definition 5.6.1 (Discrete hypergraph)**.** For a finite set X, the *discrete hypergraph* on X is

$$D(X) := (X, \varnothing).$$

Thus $D(X)$ has node set X and no hyperedges.

Definition 5.6.2 (Open Hypergraph)**.** Let X and Y be finite sets. An *open hypergraph from X to Y* is a triple

$$\mathcal{O} = (H, i, o),$$

where

$$H = (V, \mathcal{E})$$

is a finite hypergraph, and

$$i : X \to V, \qquad o : Y \to V$$

are set maps, called the *input interface map* and the *output interface map*, respectively.

Equivalently, an open hypergraph may be written as a cospan

$$D(X) \xrightarrow{i} H \xleftarrow{o} D(Y),$$

where $D(X)$ and $D(Y)$ are discrete hypergraphs.

Remark 5.6.3. *The maps i and o specify which vertices of the ambient hypergraph serve as boundary vertices. The maps need not be injective unless one explicitly imposes that restriction.*

Definition 5.6.4 (Open n-SuperHyperGraph)**.** Let V_0 be a finite nonempty base set, let $n \in \mathbb{N}_0$, and let

$$\mathrm{SHG}^{(n)} = (V, E, \partial)$$

be an n-SuperHyperGraph over V_0.

Let X and Y be finite sets. An *open n-SuperHyperGraph from X to Y* is a tuple

$$\mathcal{O}^{(n)} = (V, E, \partial, X, Y, i, o),$$

such that

1. (V, E, ∂) is an n-SuperHyperGraph over V_0;
2. $$i : X \to V$$
   is an *input interface map*;
3. $$o : Y \to V$$
   is an *output interface map.*

The elements of X and Y are called *input interface labels* and *output interface labels*, respectively. For $x \in X$, the value $i(x) \in V$ is the input boundary supervertex selected by x, and for $y \in Y$, the value $o(y) \in V$ is the output boundary supervertex selected by y.

Remark 5.6.5. *An open n-SuperHyperGraph is therefore an n-SuperHyperGraph equipped with specified input and output interfaces. It extends the notion of an open hypergraph by replacing ordinary vertices with n-supervertices.*

Remark 5.6.6 (Special case $n = 0$)**.** *When $n = 0$, one has*

$$\mathcal{P}^0(V_0) = V_0,$$

so the supervertex set V is an ordinary set of vertices. Hence an open 0-SuperHyperGraph is precisely an open hypergraph, up to the harmless use of edge identifiers and the incidence map ∂.

Theorem 5.6.7 (Well-definedness of Open n-SuperHyperGraphs)**.** *Let V_0 be a finite nonempty set and let $n \in \mathbb{N}_0$. Assume that:*

1. $$V \subseteq \mathcal{P}^n(V_0)$$
   is a finite set;
2. *E is a finite set;*
3. $$\partial : E \to \mathcal{P}^*(V)$$
   is a well-defined map;
4. *X and Y are finite sets;*
5. $$i : X \to V, \qquad o : Y \to V$$
   are well-defined maps.

Then

$$\mathcal{O}^{(n)} = (V, E, \partial, X, Y, i, o)$$

is a well-defined open n-SuperHyperGraph over V_0.

Proof. We verify each component of the definition.

First, since V_0 is a finite nonempty set and $n \in \mathbb{N}_0$, the iterated powerset $\mathcal{P}^n(V_0)$ is well-defined by finite recursion:

$$\mathcal{P}^0(V_0) = V_0, \qquad \mathcal{P}^{k+1}(V_0) = \mathcal{P}(\mathcal{P}^k(V_0)).$$

Hence the condition

$$V \subseteq \mathcal{P}^n(V_0)$$

is meaningful, and V is a well-defined finite set of n-supervertices.

Next, since V is a set, the family

$$\mathcal{P}^*(V) = \mathcal{P}(V) \setminus \{\varnothing\}$$

is also well-defined. Therefore the incidence map

$$\partial : E \to \mathcal{P}^*(V)$$

is a legitimate set-theoretic map assigning to each superedge identifier a nonempty subset of V. Consequently,

$$(V, E, \partial)$$

is a well-defined n-SuperHyperGraph over V_0.

Finally, X and Y are finite sets by assumption, and

$$i : X \to V, \qquad o : Y \to V$$

are well-defined maps into the supervertex set V. Thus the input and output interfaces are well-defined.

All components required in Definition 5.6.4 therefore exist and are compatible. Hence

$$\mathcal{O}^{(n)} = (V, E, \partial, X, Y, i, o)$$

is a well-defined open n-SuperHyperGraph. □

Corollary 5.6.8. *For every n-SuperHyperGraph (V, E, ∂) over V_0, and for every pair of finite sets X, Y equipped with maps $i : X \to V$ and $o : Y \to V$, there exists an associated open n-SuperHyperGraph*

$$(V, E, \partial, X, Y, i, o).$$

Proof. Immediate from Theorem 5.6.7. □

5.7 Combinatorial Map

A combinatorial map is a connected properly edge-colored regular graph in which each vertex is incident with exactly one edge of each color.

Definition 5.7.1 (Combinatorial map)**.** Let I be a finite nonempty set of colors. A *combinatorial map over I* is a pair

$$\mathcal{G} = (G, \tau),$$

where $G = (V, E)$ is a connected graph and

$$\tau : E \to I$$

is an edge-coloring such that, for every vertex $v \in V$ and every color $i \in I$, there exists a unique edge $e \in E$ incident with v satisfying

$$\tau(e) = i.$$

Equivalently, G is $|I|$-regular and no two distinct edges incident with the same vertex have the same color.

The integer

$$\operatorname{rank}(\mathcal{G}) := |I|$$

is called the *rank* of $\mathcal{G}$.

Definition 5.7.2 (Residues and faces)**.** Let $\mathcal{G} = (G, \tau)$ be a combinatorial map over I, and let $J \subseteq I$. Define the spanning subgraph

$$G_J := (V, \tau^{-1}(J)),$$

that is, the graph obtained from G by retaining exactly the edges whose colors belong to J. A connected component of G_J is called a *residue of type J*.

For $i \in I$, a residue of type

$$I \setminus \{i\}$$

is called an *i-face* of $\mathcal{G}$.

Remark 5.7.3. *The above definition is the standard graph-theoretic abstraction of maps on surfaces and their higher-rank analogues. In rank 3, one may think of the three colors as corresponding to the three kinds of adjacency arising from flags of a cell decomposition.*

Example 5.7.4 (A rank-3 combinatorial map associated with the tetrahedron)**.** Let

$$I = \{0, 1, 2\}, \qquad V = \{v_1, v_2, v_3, v_4\}.$$

Take G to be the complete graph K_4 on V, with edge set

$$E = \big\{\{v_1, v_2\}, \{v_3, v_4\}, \{v_1, v_4\}, \{v_2, v_3\}, \{v_1, v_3\}, \{v_2, v_4\}\big\}.$$

Define the edge-coloring $\tau : E \to I$ by

$$\tau(\{v_1, v_2\}) = \tau(\{v_3, v_4\}) = 0,$$
$$\tau(\{v_1, v_4\}) = \tau(\{v_2, v_3\}) = 1,$$
$$\tau(\{v_1, v_3\}) = \tau(\{v_2, v_4\}) = 2.$$

Then each vertex is incident with exactly one edge of color 0, one edge of color 1, and one edge of color 2. For example,

$$v_1 \text{ is incident with } \{v_1, v_2\} \text{ of color } 0,\ \{v_1, v_4\} \text{ of color } 1,\ \{v_1, v_3\} \text{ of color } 2.$$

The same property holds for v_2, v_3, v_4. Hence

$$\mathcal{G} = (G, \tau)$$

is a combinatorial map of rank 3.

Its 0-faces are the residues of type $\{1, 2\}$, its 1-faces are the residues of type $\{0, 2\}$, and its 2-faces are the residues of type $\{0, 1\}$. In this example, each such residue is a 4-cycle. For instance, the unique residue of type $\{1, 2\}$ has edge set

$$\big\{\{v_1, v_4\}, \{v_2, v_3\}, \{v_1, v_3\}, \{v_2, v_4\}\big\},$$

which forms the cycle

$$v_4 - v_1 - v_3 - v_2 - v_4.$$

An illustration is given in Fig. 5.3.

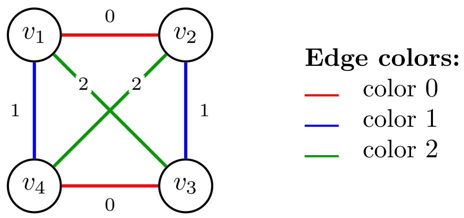


Figure 5.3: A rank-3 combinatorial map on K_4. Each vertex is incident with exactly one edge of each color $0, 1, 2$.

5.8 Cognitive HyperGraphs and Cognitive SuperHyperGraphs

In this section, we define the notions of a Cognitive HyperGraph and a Cognitive n-SuperHyperGraph as higher-order extensions of a cognitive graph. A Cognitive HyperGraph models cognitive entities as labeled vertices and multiway cognitive relations as labeled hyperedges, capturing structured group interactions semantically[258]. A Cognitive SuperHyperGraph represents higher-order cognitive groupings using iterated-powerset supervertices and labeled superhyperedges, expressing nested semantic relations among concept collections.

Definition 5.8.1 (Cognitive HyperGraph)**.** Let S be a nonempty set of distinguished locations, concepts, or cognitive units. Let L_V and L_E be nonempty label sets.

A *Cognitive HyperGraph* is a quadruple

$$\mathcal{CH} = (V, E, \ell_V, \ell_E),$$

where

1. $$V \subseteq S$$
   is a finite set of *cognitive vertices*;
2. $$E \subseteq \mathcal{P}^*(V)$$
   is a finite set of nonempty *cognitive hyperedges*;
3. $$\ell_V : V \to L_V$$
   is a *vertex-label function*;
4. $$\ell_E : E \to L_E$$
   is a *hyperedge-label function.*

Each hyperedge $e \in E$ represents a cognitively meaningful multiway relation among the vertices contained in e, while ℓ_V and ℓ_E encode semantic information attached to vertices and hyperedges, respectively.

Remark 5.8.2. *If every hyperedge has cardinality* 2*, then a Cognitive HyperGraph reduces to a labeled cognitive graph.*

Definition 5.8.3 (Cognitive n-SuperHyperGraph)**.** Let S be a nonempty base set, let $n \in \mathbb{N}_0$, and let L_V, L_E be nonempty label sets.

A *Cognitive n-SuperHyperGraph* is a quadruple

$$\mathcal{CSH}^{(n)} = (V, E, \ell_V, \ell_E),$$

such that

1. $$V \subseteq \mathcal{P}^n(S)$$
   is a finite set of *cognitive n-supervertices*;

2.
$$E \subseteq \mathcal{P}^*(V)$$
is a finite set of nonempty *cognitive n-superhyperedges*;

3.
$$\ell_V : V \to L_V$$
is a *supervertex-label function*;

4.
$$\ell_E : E \to L_E$$
is a *superhyperedge-label function.*

Thus, a Cognitive n-SuperHyperGraph is a labeled higher-order relational structure in which the vertices themselves lie at the n-th iterated powerset level over the base set S, and the superedges are nonempty subsets of the supervertex set V.

5.9 Multimodal Graph, HyperGraph, and SuperHyperGraph

A multimodal graph represents one vertex set through multiple edge modalities, capturing different interaction types or views within one network [259, 260, 261]. A multimodal hypergraph uses multiple hyperedge modalities on a shared vertex set to model higher-order relations simultaneously within one framework [262, 263, 264]. A multimodal superhypergraph organizes multiple modalities over hierarchical set-based vertices and superhyperedges, describing complex higher-order relations across abstraction levels simultaneously [246].

Definition 5.9.1 (Multimodal Graph)**.** Let V be a finite nonempty set of vertices, and let $M \in \mathbb{N}$ be the number of modalities. For each modality $m \in \{1, \dots, M\}$, let
$$G_m = (V, E_m, w_m)$$
be a weighted graph on the common vertex set V, where
$$E_m \subseteq \big\{\{u, v\} \subseteq V : u \neq v\big\}$$
is the edge set of modality m, and
$$w_m : E_m \to \mathbb{R}_{>0}$$
is a positive edge-weight function. Let
$$\alpha_1, \dots, \alpha_M \geq 0, \qquad \sum_{m=1}^{M} \alpha_m = 1,$$
be modality-combination weights.

Then the *multimodal graph* is the tuple
$$\mathcal{G} = \big(V, \{E_m\}_{m=1}^{M}, \{w_m\}_{m=1}^{M}, \{\alpha_m\}_{m=1}^{M}\big).$$

Definition 5.9.2 (Multimodal HyperGraph)**.** Let V be a finite nonempty set of vertices, and let $M \in \mathbb{N}$. For each modality $m \in \{1, \dots, M\}$, let
$$H_m = (V, \mathcal{E}_m, w_m)$$
be a weighted hypergraph on the common vertex set V, where
$$\mathcal{E}_m \subseteq \mathcal{P}^*(V) := \mathcal{P}(V) \setminus \{\emptyset\}$$
is the set of hyperedges of modality m, and
$$w_m : \mathcal{E}_m \to \mathbb{R}_{>0}$$
is a positive hyperedge-weight function. Let
$$\alpha_1, \dots, \alpha_M \geq 0, \qquad \sum_{m=1}^{M} \alpha_m = 1.$$

Then the *multimodal hypergraph* is the tuple
$$\mathcal{H} = \big(V, \{\mathcal{E}_m\}_{m=1}^{M}, \{w_m\}_{m=1}^{M}, \{\alpha_m\}_{m=1}^{M}\big).$$

Definition 5.9.3 (Multimodal n-SuperHyperGraph)**.** [246] Let V_0 be a finite nonempty base set, and let $n \in \mathbb{N} \cup \{0\}$. Define the iterated powersets recursively by

$$\mathcal{P}^0(V_0) := V_0, \qquad \mathcal{P}^{k+1}(V_0) := \mathcal{P}\big(\mathcal{P}^k(V_0)\big) \quad (k \geq 0).$$

Let

$$V \subseteq \mathcal{P}^n(V_0)$$

be a finite set of n-supervertices, and let $M \in \mathbb{N}$. For each modality $m \in \{1, \ldots, M\}$, let

$$\mathcal{H}_m^{(n)} = (V, \mathcal{E}_m^{(n)}, w_m)$$

be a weighted hypergraph on the common set V, where

$$\mathcal{E}_m^{(n)} \subseteq \mathcal{P}^*(V)$$

is the set of n-superhyperedges of modality m, and

$$w_m : \mathcal{E}_m^{(n)} \to \mathbb{R}_{>0}$$

is a positive weight function. Let

$$\alpha_1, \ldots, \alpha_M \geq 0, \qquad \sum_{m=1}^{M} \alpha_m = 1.$$

Then the *multimodal n-SuperHyperGraph* is the tuple

$$\mathcal{H}^{(n)} = \big(V, \{\mathcal{E}_m^{(n)}\}_{m=1}^M, \{w_m\}_{m=1}^M, \{\alpha_m\}_{m=1}^M\big).$$

Example 5.9.4 (A concrete Multimodal 1-SuperHyperGraph)**.** Let the base set be

$$V_0 = \{a, b, c, d, e\},$$

and take $n = 1$. Then

$$\mathcal{P}^1(V_0) = \mathcal{P}(V_0),$$

so each 1-supervertex is a subset of V_0.

Define the common set of 1-supervertices by

$$V = \{v_1, v_2, v_3, v_4\} \subseteq \mathcal{P}(V_0),$$

where

$$v_1 = \{a, b\}, \qquad v_2 = \{b, c, d\}, \qquad v_3 = \{e\}, \qquad v_4 = \{a, e\}.$$

Thus each supervertex represents a group of base elements.

Now let the number of modalities be

$$M = 2.$$

Modality 1: communication. Define the weighted hypergraph

$$\mathcal{H}_1^{(1)} = (V, \mathcal{E}_1^{(1)}, w_1)$$

by

$$\mathcal{E}_1^{(1)} = \{E_1^{(1)}, E_2^{(1)}\},$$

where

$$E_1^{(1)} = \{v_1, v_2\}, \qquad E_2^{(1)} = \{v_2, v_3\},$$

and assign positive weights

$$w_1\Big(E_1^{(1)}\Big) = 0.8, \qquad w_1\Big(E_2^{(1)}\Big) = 0.6.$$

Modality 2: resource sharing. Define the weighted hypergraph

$$\mathcal{H}_2^{(1)} = (V, \mathcal{E}_2^{(1)}, w_2)$$

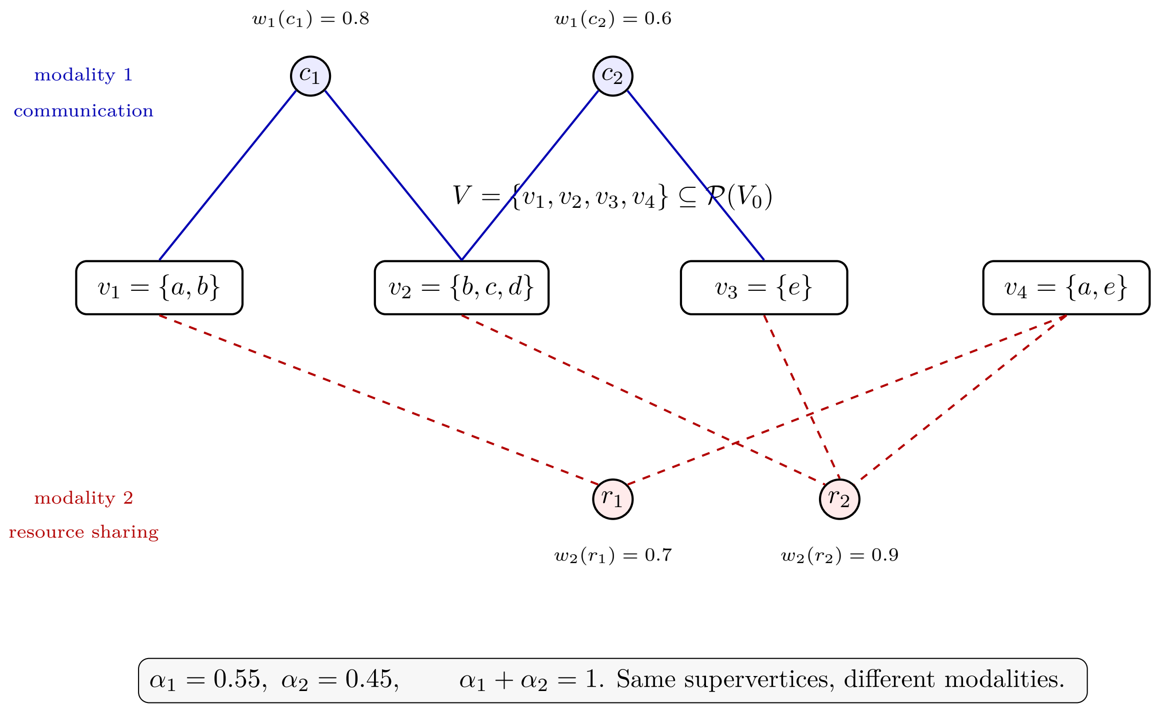


Figure 5.4: A concrete Multimodal 1-SuperHyperGraph. The same set of 1-supervertices is shared by two modalities: communication (solid blue) and resource sharing (dashed red).

by

$$\mathcal{E}_2^{(1)} = \{E_1^{(2)}, E_2^{(2)}\},$$

where

$$E_1^{(2)} = \{v_1, v_4\}, \qquad E_2^{(2)} = \{v_2, v_3, v_4\},$$

and assign positive weights

$$w_2\Big(E_1^{(2)}\Big) = 0.7, \qquad w_2\Big(E_2^{(2)}\Big) = 0.9.$$

Finally, choose modality-combination weights

$$\alpha_1 = 0.55, \qquad \alpha_2 = 0.45,$$

so that

$$\alpha_1, \alpha_2 \geq 0, \qquad \alpha_1 + \alpha_2 = 1.$$

Therefore,

$$\mathcal{H}^{(1)} = \Big(V, \{\mathcal{E}_1^{(1)}, \mathcal{E}_2^{(1)}\}, \{w_1, w_2\}, \{\alpha_1, \alpha_2\}\Big)$$

is a Multimodal 1-SuperHyperGraph in the sense of the above definition.

The common supervertex set V is shared by both modalities. The first modality describes communication relations among groups, while the second modality describes resource-sharing relations among the same groups. Hence the same hierarchical vertex system is observed through multiple distinct interaction modes. An illustration is given in Fig. 5.4.

5.10 Operadic Interaction Graph (OIG)

An Operadic Interaction Graph is a colored operad whose operations represent typed multi-input interactions; operad composition models gluing interactions, encoding higher-order, compositional network structure.

Definition 5.10.1 (Operadic Interaction Graph (OIG))**.** Let C be a set of *colors* (types). An *Operadic Interaction Graph* is a symmetric C-colored operad

$$\mathsf{OIG} = \mathcal{O} = \big(\mathcal{O}(c_1, \ldots, c_n; c)\big)_{n \geq 0,\ (c_1, \ldots, c_n; c) \in C^n \times C}$$

consisting of:

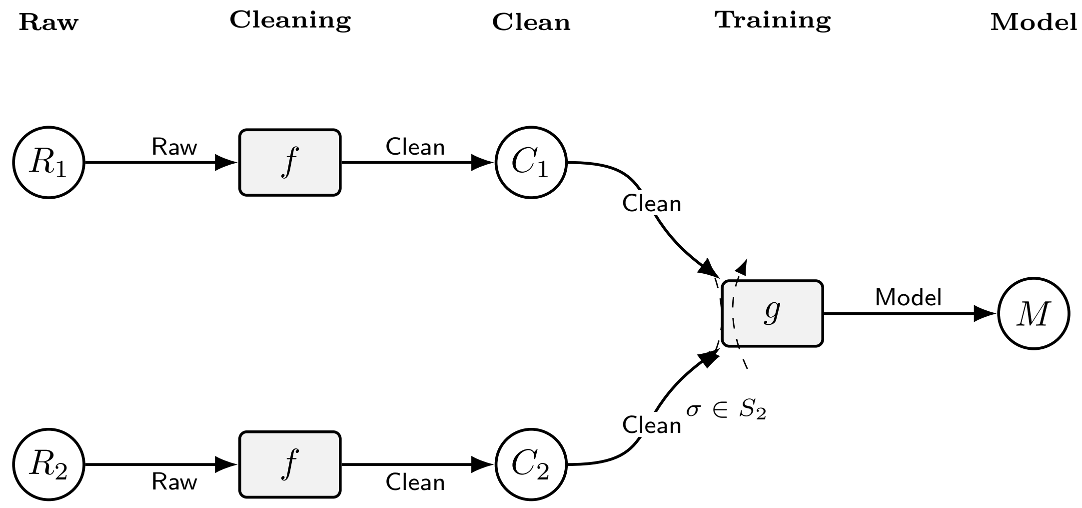
1. operation sets $\mathcal{O}(c_1, \ldots, c_n; c)$ for all $n \geq 0$ and $(c_1, \ldots, c_n; c) \in C^n \times C$;
2. right actions of the symmetric groups, i.e. for each $\pi \in S_n$,
$$(-) \cdot \pi : \mathcal{O}(c_1, \ldots, c_n; c) \to \mathcal{O}(c_{\pi(1)}, \ldots, c_{\pi(n)}; c);$$
3. substitution (composition) maps
$$\gamma : \ \mathcal{O}(c_1, \ldots, c_n; c) \times \prod_{i=1}^{n} \mathcal{O}(d_{i1}, \ldots, d_{ik_i}; c_i) \to \mathcal{O}(d_{11}, \ldots, d_{1k_1}, \ldots, d_{n1}, \ldots, d_{nk_n}; c);$$
4. units $\mathrm{id}_c \in \mathcal{O}(c; c)$ for all $c \in C$,

satisfying the standard operad axioms (associativity of substitution, unitality, and equivariance with respect to the S_n-actions). Operations are interpreted as higher-arity interaction primitives, and γ encodes compositional gluing of interactions.

Example 5.10.2 (An Operadic Interaction Graph for typed workflows)**.** Let the color set be
$$C = \{\mathsf{Raw}, \mathsf{Clean}, \mathsf{Model}\},$$
interpreted as data/product types in a workflow. Define a symmetric C-colored operad $\mathcal{O}$ by specifying the following generating operations:
$$f \in \mathcal{O}(\mathsf{Raw}; \mathsf{Clean}) \quad \text{(cleaning)}, \qquad g \in \mathcal{O}(\mathsf{Clean}, \mathsf{Clean}; \mathsf{Model}) \quad \text{(training from two datasets)}.$$
Include all operations obtained from these generators by operadic substitution and by the symmetric actions (permuting the two inputs of g), together with the required units
$$\mathrm{id}_{\mathsf{Raw}} \in \mathcal{O}(\mathsf{Raw}; \mathsf{Raw}), \quad \mathrm{id}_{\mathsf{Clean}} \in \mathcal{O}(\mathsf{Clean}; \mathsf{Clean}), \quad \mathrm{id}_{\mathsf{Model}} \in \mathcal{O}(\mathsf{Model}; \mathsf{Model}).$$
For instance, the composite operation
$$h \ = \ \gamma\big(g;\ f, f\big) \ \in \ \mathcal{O}(\mathsf{Raw}, \mathsf{Raw}; \mathsf{Model})$$
represents the workflow "clean two raw datasets and then train a model from the two cleaned datasets." With these operation sets, symmetric actions, units, and substitution maps, $\mathcal{O}$ forms a symmetric C-colored operad, and hence defines an Operadic Interaction Graph in the sense of Definition 5.10.1. An overview diagram of this example is provided in Fig. 5.5.

Theorem 5.10.3 (Well-definedness of Operadic Interaction Graph semantics)**.** *Let C be a set of colors, and let*
$$\mathsf{OIG} = \mathcal{O} = \big(\mathcal{O}(c_1, \ldots, c_n; c)\big)_{n \geq 0,\ (c_1, \ldots, c_n; c) \in C^n \times C}$$
be the data of Definition 5.10.1, together with symmetric-group actions, substitution maps, and units, satisfying the colored operad axioms (associativity, unitality, and equivariance). Then the notion of an Operadic Interaction Graph *is well-defined. More precisely:*

1. *For every profile $(c_1, \ldots, c_n; c) \in C^n \times C$, the set $\mathcal{O}(c_1, \ldots, c_n; c)$ is a well-typed set of n-ary operations with input colors $c_1, \ldots, c_n$ and output color c.*
2. *For every $\pi \in S_n$, the action map*
$$(-) \cdot \pi : \mathcal{O}(c_1, \ldots, c_n; c) \to \mathcal{O}(c_{\pi(1)}, \ldots, c_{\pi(n)}; c)$$
is well-defined and preserves output color while permuting input colors.
3. *For every choice of profiles*
$$g \in \mathcal{O}(c_1, \ldots, c_n; c), \qquad f_i \in \mathcal{O}(d_{i1}, \ldots, d_{ik_i}; c_i) \quad (1 \leq i \leq n),$$
the substitution
$$\gamma(g; f_1, \ldots, f_n)$$
is well-defined and lies in
$$\mathcal{O}(d_{11}, \ldots, d_{1k_1}, \ldots, d_{n1}, \ldots, d_{nk_n}; c).$$



Typed operations: $f \in \mathcal{O}(\mathsf{Raw}; \mathsf{Clean})$ (cleaning), $g \in \mathcal{O}(\mathsf{Clean}, \mathsf{Clean}; \mathsf{Model})$ (training).
Operadic composition: $h = \gamma(g; f, f) \in \mathcal{O}(\mathsf{Raw}, \mathsf{Raw}; \mathsf{Model})$. Interpretation: clean two raw datasets, then train a model from the two cleaned datasets.

Figure 5.5: An Operadic Interaction Graph for a typed workflow. The composite workflow is the operadic substitution $h = \gamma(g; f, f)$.

4. *Any formal interaction expression built from operations in $\mathcal{O}$, units, symmetric actions, and iterated substitutions has a well-defined typed value, and its value is independent of the order of evaluation (parenthesization of substitutions) and coherent permutation choices, by the operad axioms.*

Hence the interpretation of operations as higher-arity interaction primitives and of γ as compositional gluing is mathematically well-posed.

Proof. We verify each point.

(1) Well-typed operation sets. By assumption, for each $n \geq 0$ and each profile $(c_1, \ldots, c_n; c) \in C^n \times C$, a set $\mathcal{O}(c_1, \ldots, c_n; c)$ is specified. Thus every element $x \in \mathcal{O}(c_1, \ldots, c_n; c)$ has a definite type: it is an n-ary operation with ordered input colors $c_1, \ldots, c_n$ and output color c. This is exactly the required typing data.

(2) Symmetric actions are well-defined. Fix $n \geq 0$, a profile $(c_1, \ldots, c_n; c)$, and $\pi \in S_n$. By assumption, there is a map

$$(-) \cdot \pi : \mathcal{O}(c_1, \ldots, c_n; c) \to \mathcal{O}(c_{\pi(1)}, \ldots, c_{\pi(n)}; c).$$

Since $(c_{\pi(1)}, \ldots, c_{\pi(n)}) \in C^n$, the codomain is a valid operation set. Hence the action is well-defined as a type-preserving reindexing of the inputs (with the same output color c). For $n = 0$, this is also well-defined because S_0 is the trivial group.

(3) Substitution is well-defined and correctly typed. Let

$$g \in \mathcal{O}(c_1, \ldots, c_n; c), \qquad f_i \in \mathcal{O}(d_{i1}, \ldots, d_{ik_i}; c_i) \quad (1 \leq i \leq n).$$

The output color of each f_i is c_i, which matches the i-th input color of g. Therefore the tuple $(g; f_1, \ldots, f_n)$ is composable.

Now consider the concatenated list of colors

$$(d_{11}, \ldots, d_{1k_1}, \ldots, d_{n1}, \ldots, d_{nk_n}).$$

Each entry lies in C, so this is an element of $C^{k_1+\cdots+k_n}$. Hence the target set

$$\mathcal{O}(d_{11}, \ldots, d_{1k_1}, \ldots, d_{n1}, \ldots, d_{nk_n}; c)$$

is a legitimate component of the colored operad.

By assumption, the substitution map

$$\gamma : \mathcal{O}(c_1, \ldots, c_n; c) \times \prod_{i=1}^{n} \mathcal{O}(d_{i1}, \ldots, d_{ik_i}; c_i) \to \mathcal{O}(d_{11}, \ldots, d_{1k_1}, \ldots, d_{n1}, \ldots, d_{nk_n}; c)$$

is defined, so $\gamma(g; f_1, \ldots, f_n)$ exists and has the stated type. Therefore substitution is well-defined.

(4) Formal operadic expressions have well-defined values. A formal interaction expression is built inductively from:

- basic operations $x \in \mathcal{O}(c_1, \ldots, c_n; c)$,
- units $\mathrm{id}_c \in \mathcal{O}(c; c)$,
- symmetric actions $x \mapsto x \cdot \pi$,
- substitutions $\gamma(-; -, \ldots, -)$.

By (1)–(3), each construction step is well-typed whenever the colors match, so every such formal expression has a well-defined output color and a well-defined ordered list of input colors.

It remains to show independence of evaluation order and coherent permutations. This follows from the operad axioms:

- *Associativity of substitution* implies that iterated substitutions give the same result regardless of how one parenthesizes the gluing process.
- *Unitality* implies that inserting or removing the units id_c does not change the operation.
- *Equivariance* implies compatibility between substitutions and symmetric-group actions, so permuting inputs before or after substitution yields the same result in the prescribed sense.

Therefore the semantic value of a formal operadic interaction expression is uniquely determined by the expression modulo the standard operad identifications.

This proves that the OIG construction is well-defined. □

5.11 Symmetric Monoidal Wiring Graph (SMWG)

A Symmetric Monoidal Wiring Graph models networks as morphisms in a symmetric monoidal category: generators are multiport components; composition and tensor encode wiring and parallelism.

Definition 5.11.1 (Symmetric Monoidal Wiring Graph (SMWG))**.** Let T be a set of *port types.* A *Symmetric Monoidal Wiring Graph* is a triple

$$\mathsf{SMWG} = (\mathcal{C}, \otimes, \mathcal{G}),$$

where $(\mathcal{C}, \otimes, \mathbb{I})$ is a symmetric monoidal category whose objects are generated by T under $\otimes$ (so objects are tensor words $t_1 \otimes \cdots \otimes t_m$ with $t_i \in T$), and $\mathcal{G}$ is a specified set of *generating morphisms* (components)

$$f : \; t_1 \otimes \cdots \otimes t_m \longrightarrow s_1 \otimes \cdots \otimes s_n \qquad (t_i, s_j \in T).$$

A *network* in SMWG is any morphism in $\mathcal{C}$, built from $\mathcal{G}$ using categorical composition (sequential wiring) and $\otimes$ (parallel composition), modulo the coherence isomorphisms of the symmetric monoidal structure.

Example 5.11.2 (A Symmetric Monoidal Wiring Graph for simple digital circuits)**.** Let the set of port types be

$$T = \{\mathsf{Bit}\}.$$

Let $\mathcal{C}$ be the free symmetric monoidal category generated by T together with the following generating morphisms (circuit components):

$$\mathrm{AND} : \mathsf{Bit} \otimes \mathsf{Bit} \to \mathsf{Bit}, \qquad \mathrm{NOT} : \mathsf{Bit} \to \mathsf{Bit}, \qquad \mathrm{COPY} : \mathsf{Bit} \to \mathsf{Bit} \otimes \mathsf{Bit}.$$

Set

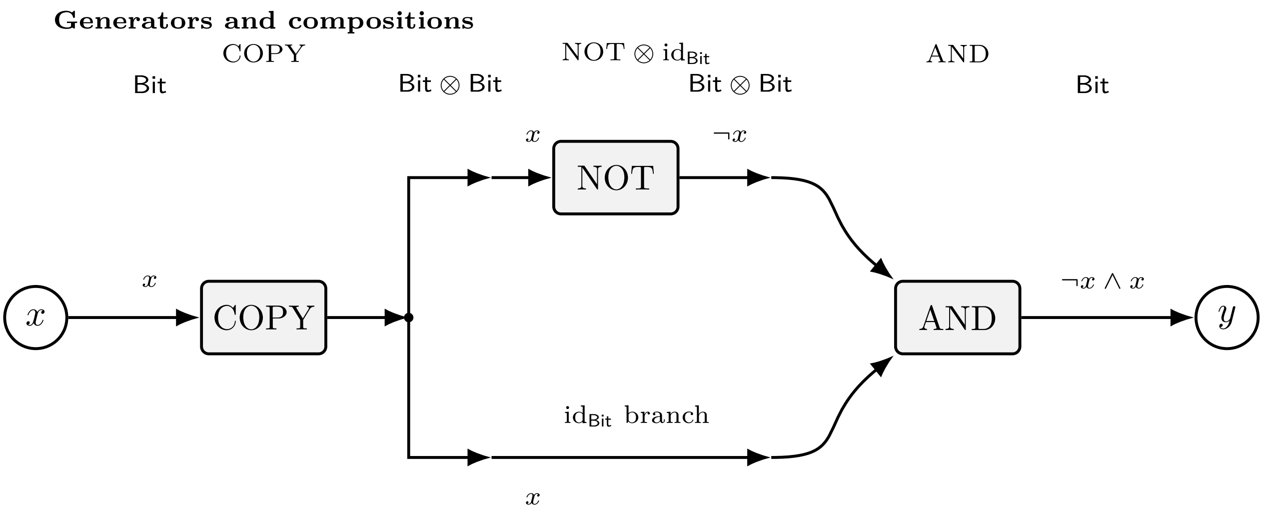
$$\mathcal{G} = \{\mathrm{AND}, \mathrm{NOT}, \mathrm{COPY}\}.$$

Then objects of $\mathcal{C}$ are tensor words $\mathsf{Bit}^{\otimes m}$, and morphisms are wiring diagrams built from the generators by sequential composition and parallel composition $\otimes$ (with swaps provided by the symmetry).

For instance, the network

$$F \;=\; \mathrm{AND} \circ \big(\mathrm{NOT} \otimes \mathrm{id}_{\mathsf{Bit}}\big) \circ \mathrm{COPY} \;:\; \mathsf{Bit} \longrightarrow \mathsf{Bit}$$

represents the circuit that takes an input bit x, copies it to (x, x), negates the first copy to $(\neg x, x)$, and then applies AND, producing $\neg x \wedge x$. Hence $(\mathcal{C}, \otimes, \mathcal{G})$ is a Symmetric Monoidal Wiring Graph in the sense of Definition 5.11.1. An overview diagram of this example is provided in Fig. 5.6.



SMWG interpretation. This diagram represents the morphism

$$F = \text{AND} \circ \big(\text{NOT} \otimes \text{id}_{\mathsf{Bit}}\big) \circ \text{COPY} : \mathsf{Bit} \to \mathsf{Bit}.$$

It is built from the generators COPY, NOT, AND using sequential composition and tensor (parallel) composition. The symmetric monoidal structure also provides wire swaps (symmetry), although no swap is needed in this specific circuit.

Figure 5.6: A Symmetric Monoidal Wiring Graph for a simple digital circuit. The circuit copies an input bit, negates one copy, and combines the two signals via AND.

Theorem 5.11.3 (Well-definedness of Symmetric Monoidal Wiring Graph semantics)**.** *Let T be a set of port types, and let*

$$\mathsf{SMWG} = (\mathcal{C}, \otimes, \mathcal{G})$$

be as in Definition 5.11.1, where $(\mathcal{C}, \otimes, \mathbb{I})$ is a symmetric monoidal category and $\mathcal{G}$ is a set of generating morphisms

$$f : \ t_1 \otimes \cdots \otimes t_m \longrightarrow s_1 \otimes \cdots \otimes s_n \qquad (t_i, s_j \in T).$$

Then the notion of a network *in* SMWG *is well-defined:*

1. *Every formal wiring expression built from generators in $\mathcal{G}$, identity morphisms, sequential composition ($\circ$), tensor product ($\otimes$), and symmetry maps (swaps) is well-typed, provided the source/target objects match at each composition step.*
2. *Such a formal expression determines a unique morphism in $\mathcal{C}$.*
3. *If two formal expressions differ only by parenthesization, insertion/removal of unit objects $\mathbb{I}$, or by replacing canonical rebracketing/symmetry isomorphisms using the symmetric monoidal coherence axioms, then they define the same network (i.e. the same morphism in $\mathcal{C}$, up to the canonical coherence identifications).*

In particular, "networks built from $\mathcal{G}$ by sequential and parallel wiring, modulo coherence" is a mathematically well-defined notion.

Proof. We prove the three claims.

(1) Well-typed formation of wiring expressions. By assumption, every generator $f \in \mathcal{G}$ is already a morphism in $\mathcal{C}$ with a specified source and target object, each of which is a tensor word in the types T.

The class of formal wiring expressions is generated inductively from:

- generators $f \in \mathcal{G}$,
- identity morphisms $\text{id}_X : X \to X$ for objects $X \in \text{Ob}(\mathcal{C})$,
- composition $g \circ f$ whenever $\text{cod}(f) = \text{dom}(g)$,

- tensor $f \otimes g$ whenever $f : X \to Y$ and $g : X' \to Y'$,
- symmetry isomorphisms $\sigma_{X,Y} : X \otimes Y \to Y \otimes X$.

Since $\mathcal{C}$ is a category with a monoidal product, each of these constructions is defined whenever the indicated typing conditions hold. Therefore every inductively formed wiring expression is well-typed.

(2) Every well-typed expression denotes a morphism in $\mathcal{C}$. We define the semantics of formal expressions by structural recursion:

- a generator $f \in \mathcal{G}$ is interpreted as itself (a morphism in $\mathcal{C}$);
- id_X is interpreted as the identity on X;
- $g \circ f$ is interpreted as the categorical composite of the interpretations of f and g;
- $f \otimes g$ is interpreted as the monoidal product of the interpretations;
- $\sigma_{X,Y}$ is interpreted as the symmetry morphism in $\mathcal{C}$.

Because $(\mathcal{C}, \otimes, \mathbb{I})$ is a symmetric monoidal category, all these operations exist and preserve typing. Hence every well-typed formal wiring expression evaluates to a unique morphism in $\mathcal{C}$.

(3) Independence of parenthesization and coherence choices. A potential ambiguity arises because tensor words may be written with different parenthesizations (and unit insertions), e.g.

$$(t_1 \otimes t_2) \otimes t_3 \qquad \text{vs.} \qquad t_1 \otimes (t_2 \otimes t_3),$$

and because one may insert canonical associators, unitors, and symmetries at intermediate steps.

However, in a symmetric monoidal category, the coherence theorem (Mac Lane coherence, together with symmetry coherence) implies that all canonical composites built from associators, unitors, and symmetries between the same tensor expressions coincide. Equivalently, canonical rebracketing/reordering maps are uniquely determined by the source and target tensor words.

Therefore, if two formal wiring expressions differ only by:

- parenthesization of tensor products,
- insertion/removal of $\mathbb{I}$ via unitors,
- replacement by canonically coherent associativity/unit/symmetry isomorphisms,

then their interpretations in $\mathcal{C}$ agree (up to the canonical identifications specified by coherence). Hence the equivalence class "modulo coherence" has a unique semantic value.

This proves that the notion of a network in SMWG is well-defined. □

5.12 Relational-Arity Graph (RAG)

A Relational-Arity Graph is a finite relational structure: vertices with relations of varying arities, so each k-ary relation encodes k-way interactions, possibly weighted.

Definition 5.12.1 (Relational-Arity Graph (RAG))**.** A *Relational-Arity Graph* is a pair

$$\mathsf{RAG} = (\Sigma, \mathfrak{G}),$$

where $\Sigma = \{R_\alpha \mid \alpha \in A\}$ is a relational signature with arities $\mathrm{ar}(R_\alpha) = k_\alpha \in \mathbb{N}$, and

$$\mathfrak{G} = \big(V, (R_\alpha^{\mathfrak{G}})_{\alpha \in A}\big)$$

is a Σ-structure on a vertex set V, i.e.

$$R_\alpha^{\mathfrak{G}} \subseteq V^{k_\alpha} \quad \text{for all } \alpha \in A.$$

Thus each relation symbol R_α encodes k_α-way interactions (ordered tuples).

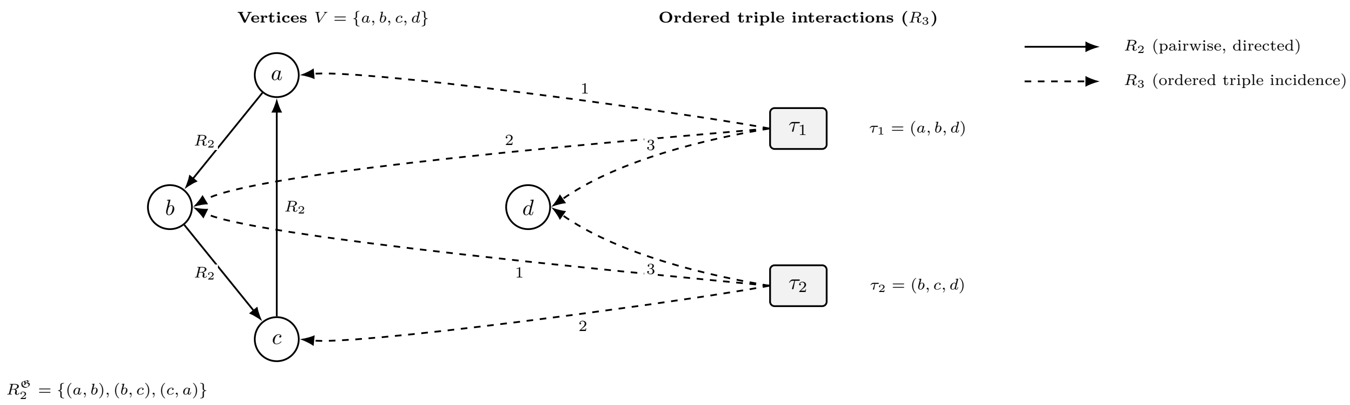


Figure 5.7: A Relational-Arity Graph (RAG) for Example 5.12.2. Solid arrows represent the binary relation $R_2^{\mathfrak{G}}$, and dashed arrows from tuple-nodes τ_1, τ_2 encode the ordered triples in $R_3^{\mathfrak{G}}$.

Example 5.12.2 (A Relational-Arity Graph with pairwise and triple interactions)**.** Let the vertex set be

$$V = \{a, b, c, d\}.$$

Consider the relational signature

$$\Sigma = \{R_2, R_3\}, \qquad \mathrm{ar}(R_2) = 2, \ \ \mathrm{ar}(R_3) = 3,$$

where R_2 is intended to encode directed pairwise interactions and R_3 encodes ordered triple interactions. Define the Σ-structure

$$\mathfrak{G} = \big(V,\ R_2^{\mathfrak{G}},\ R_3^{\mathfrak{G}}\big)$$

by specifying the interpreted relations:

$$R_2^{\mathfrak{G}} = \{(a, b), (b, c), (c, a)\} \subseteq V^2,$$

$$R_3^{\mathfrak{G}} = \{(a, b, d), (b, c, d)\} \subseteq V^3.$$

Then $\mathsf{RAG} = (\Sigma, \mathfrak{G})$ is a Relational-Arity Graph in the sense of Definition 5.12.1. Here $R_2^{\mathfrak{G}}$ records a directed 3-cycle among (a, b, c), while $R_3^{\mathfrak{G}}$ records two distinct 3-way interactions involving the vertex d. An overview diagram of this example is provided in Fig. 5.7.

Theorem 5.12.3 (Well-definedness of Relational-Arity Graphs)**.** *Let*

$$\Sigma = \{R_\alpha \mid \alpha \in A\}$$

be a relational signature together with an arity map

$$\mathrm{ar} : A \to \mathbb{N}, \qquad \alpha \mapsto k_\alpha := \mathrm{ar}(R_\alpha).$$

Let V be a set, and suppose that for each $\alpha \in A$ a relation

$$R_\alpha^{\mathfrak{G}} \subseteq V^{k_\alpha}$$

is given. Then the pair

$$\mathsf{RAG} = (\Sigma, \mathfrak{G}), \qquad \mathfrak{G} = \big(V, (R_\alpha^{\mathfrak{G}})_{\alpha \in A}\big),$$

is a well-defined Relational-Arity Graph in the sense of Definition 5.12.1.
Moreover, for each $\alpha \in A$, membership

$$(v_1, \ldots, v_{k_\alpha}) \in R_\alpha^{\mathfrak{G}}$$

is a well-posed statement for all $(v_1, \ldots, v_{k_\alpha}) \in V^{k_\alpha}$, and thus each $R_\alpha^{\mathfrak{G}}$ defines a k_α-ary interaction on V (ordered unless additional symmetry is imposed).

Proof. To show that $(\Sigma, \mathfrak{G})$ is well-defined, we must check that every piece of data appearing in Definition 5.12.1 is properly typed.

(1) The signature is well-formed. By assumption, $\Sigma = \{R_\alpha \mid \alpha \in A\}$ is a family of relation symbols indexed by A, and each symbol R_α is assigned a positive integer arity $k_\alpha \in \mathbb{N}$. Hence the typed signature data (Σ, ar) is well-defined.

(2) Cartesian powers are well-defined. For each $\alpha \in A$, since $k_\alpha \in \mathbb{N}$, the Cartesian power

$$V^{k_\alpha} = \underbrace{V \times \cdots \times V}_{k_\alpha \text{ factors}}$$

is well-defined. Its elements are exactly ordered k_α-tuples of vertices in V.

(3) Interpreted relations are well-typed. Again by assumption, for each $\alpha \in A$,

$$R_\alpha^{\mathfrak{G}} \subseteq V^{k_\alpha}.$$

Therefore $R_\alpha^{\mathfrak{G}}$ is a k_α-ary relation on V, i.e. a set of ordered k_α-tuples. Hence the family $(R_\alpha^{\mathfrak{G}})_{\alpha \in A}$ is a well-defined interpretation of the signature Σ on V.

(4) The Σ-structure is well-defined. Combining (1)–(3), the object

$$\mathfrak{G} = \big(V, (R_\alpha^{\mathfrak{G}})_{\alpha \in A}\big)$$

is a valid Σ-structure (in the standard model-theoretic sense). Therefore the pair

$$(\Sigma, \mathfrak{G})$$

is a well-defined Relational-Arity Graph.

The final claim is immediate: for each $\alpha \in A$, because $R_\alpha^{\mathfrak{G}} \subseteq V^{k_\alpha}$, the statement

$$(v_1, \ldots, v_{k_\alpha}) \in R_\alpha^{\mathfrak{G}}$$

is meaningful exactly for $(v_1, \ldots, v_{k_\alpha}) \in V^{k_\alpha}$, and this records a k_α-way interaction. The tuples are ordered by default because V^{k_α} is an ordered Cartesian product. □

5.13 Closure-Implication Graph (CIG)

A *Closure-Implication Graph (CIG)* is a set with a closure operator; forcing relations $a \in \mathrm{cl}(S) \setminus S$ capture higher-order implications, with minimal generators encoding interactions.

Definition 5.13.1 (Closure-Implication Graph (CIG))**.** A *Closure-Implication Graph* is a pair

$$\mathsf{CIG} = (V, \mathrm{cl}),$$

where V is a set and $\mathrm{cl} : \mathcal{P}(V) \to \mathcal{P}(V)$ is a closure operator, i.e. for all $S, T \subseteq V$:

$$S \subseteq \mathrm{cl}(S), \qquad S \subseteq T \Rightarrow \mathrm{cl}(S) \subseteq \mathrm{cl}(T), \qquad \mathrm{cl}(\mathrm{cl}(S)) = \mathrm{cl}(S).$$

A higher-order forcing event is expressed by $a \in \mathrm{cl}(S) \setminus S$; minimal such S (by inclusion) may be regarded as the basic implicational generators of a.

Example 5.13.2 (A Closure-Implication Graph generated by simple rules)**.** Let

$$V = \{a, b, c, d\}.$$

Consider the following implicational rules:

$$\{a, b\} \Rightarrow c, \qquad \{c\} \Rightarrow d.$$

Define $\mathrm{cl} : \mathcal{P}(V) \to \mathcal{P}(V)$ by letting $\mathrm{cl}(S)$ be the smallest subset of V that contains S and is closed under the above rules; equivalently, $\mathrm{cl}(S)$ is obtained by repeatedly adding c whenever $\{a, b\} \subseteq S$ and adding d whenever $c \in S$, until no new elements can be added.

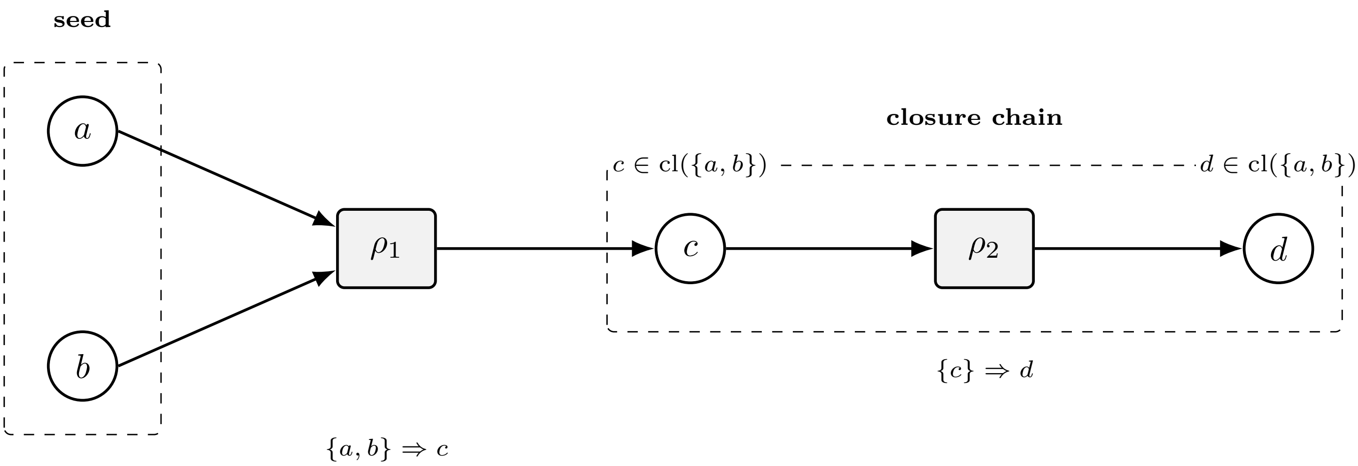


Figure 5.8: A Closure-Implication Graph generated by the rules $\{a, b\} \Rightarrow c$ and $\{c\} \Rightarrow d$.

For example,

$$\mathrm{cl}(\{a\}) = \{a\}, \qquad \mathrm{cl}(\{a, b\}) = \{a, b, c, d\}, \qquad \mathrm{cl}(\{b, c\}) = \{b, c, d\}.$$

Then cl is extensive, monotone, and idempotent, hence (V, cl) is a Closure-Implication Graph in the sense of Definition 5.13.1. In particular, $\{a, b\}$ forces c since

$$c \in \mathrm{cl}(\{a, b\}) \setminus \{a, b\},$$

and $\{a, b\}$ also forces d via the intermediate implication $\{a, b\} \Rightarrow c$ and then $\{c\} \Rightarrow d$. An overview diagram of this example is provided in Fig. 5.8.

Theorem 5.13.3 (Well-definedness of Closure-Implication Graph semantics)**.** *Let V be a set and let $\mathrm{cl} : \mathcal{P}(V) \to \mathcal{P}(V)$ be a closure operator, i.e. for all $S, T \subseteq V$,*

$$S \subseteq \mathrm{cl}(S), \qquad S \subseteq T \Rightarrow \mathrm{cl}(S) \subseteq \mathrm{cl}(T), \qquad \mathrm{cl}(\mathrm{cl}(S)) = \mathrm{cl}(S).$$

Then the pair $\mathsf{CIG} = (V, \mathrm{cl})$ is well-defined as a Closure-Implication Graph.

Moreover, for each $a \in V$, define the forcing family

$$\mathcal{F}_a := \{ S \subseteq V \setminus \{a\} \mid a \in \mathrm{cl}(S) \}.$$

Then:

1. *the higher-order forcing relation*
$$S \rightsquigarrow a \quad \Longleftrightarrow \quad a \in \mathrm{cl}(S) \setminus S$$
is well-defined on $\mathcal{P}(V) \times V$;

2. *$\mathcal{F}_a$ is upward closed under inclusion, i.e. if $S \in \mathcal{F}_a$ and $S \subseteq T \subseteq V \setminus \{a\}$, then $T \in \mathcal{F}_a$;*

3. *if V is finite, then $\mathcal{F}_a$ has inclusion-minimal elements (possibly none), and hence the "minimal implicational generators" of a are well-defined as*
$$\mathrm{MinGen}(a) := \min_{\subseteq}(\mathcal{F}_a).$$
In fact, for every $S \in \mathcal{F}_a$, there exists $M \in \mathrm{MinGen}(a)$ such that $M \subseteq S$.

Proof. The first statement is immediate: by assumption, cl is a map $\mathcal{P}(V) \to \mathcal{P}(V)$ satisfying the closure axioms, so (V, cl) is exactly a Closure-Implication Graph by Definition 5.13.1.

For (1), let $S \subseteq V$ and $a \in V$. Since $\mathrm{cl}(S) \subseteq V$, the expression

$$a \in \mathrm{cl}(S) \setminus S$$

is a meaningful truth-valued statement. Hence $S \rightsquigarrow a$ defines a well-posed binary relation between subsets $S \subseteq V$ and vertices $a \in V$.

For (2), suppose $S \in \mathcal{F}_a$ and $S \subseteq T \subseteq V \setminus \{a\}$. By $S \in \mathcal{F}_a$, we have $a \in \mathrm{cl}(S)$. By monotonicity of cl,

$$\mathrm{cl}(S) \subseteq \mathrm{cl}(T),$$

so $a \in \mathrm{cl}(T)$. Since $T \subseteq V \setminus \{a\}$, we also have $a \notin T$. Therefore $T \in \mathcal{F}_a$. Thus $\mathcal{F}_a$ is upward closed.

For (3), assume V is finite. Then $\mathcal{P}(V \setminus \{a\})$ is a finite poset under inclusion, and $\mathcal{F}_a \subseteq \mathcal{P}(V \setminus \{a\})$ is a (finite) subset of this poset. Every finite poset subset has minimal elements, so $\min_{\subseteq}(\mathcal{F}_a)$ exists (possibly empty if $\mathcal{F}_a = \varnothing$). Hence $\mathrm{MinGen}(a)$ is well-defined.

Finally, let $S \in \mathcal{F}_a$. Consider the finite set

$$\{\, T \in \mathcal{F}_a \mid T \subseteq S \,\}.$$

It is nonempty because S itself belongs to it. Choose an inclusion-minimal element M of this set. Then $M \in \mathcal{F}_a$, $M \subseteq S$, and no proper subset of M belongs to $\mathcal{F}_a$; hence $M \in \mathrm{MinGen}(a)$. This proves the last claim. □

5.14 Coalgebraic Nested-Neighborhood Graph (CNNG)

A *Coalgebraic Nested-Neighborhood Graph (CNNG)* is an F-coalgebra (V, γ) where $\gamma : V \to F(V)$ assigns nested neighborhoods, capturing iterated higher-order adjacency structure.

Definition 5.14.1 (Coalgebraic Nested-Neighborhood Graph (CNNG))**.** Let $F : \mathbf{Set} \to \mathbf{Set}$ be an endofunctor. A *Coalgebraic Nested-Neighborhood Graph* is an F-coalgebra

$$\mathsf{CNNG} = (V, \gamma), \qquad \gamma : V \to F(V).$$

In particular, letting $\mathcal{P}_f(V)$ denote the set of finite subsets of V, for any $r \geq 1$ one may take

$$F(V) = \mathcal{P}_f^r(V),$$

so that $\gamma(v) \in \mathcal{P}_f^r(V)$ assigns to each vertex v an r-nested finite neighborhood. The case $r = 1$ recovers ordinary (directed) neighborhood graphs, while $r \geq 2$ encodes iterated (neighborhood-of-neighborhood) structure.

Example 5.14.2 (A CNNG with 2-nested neighborhoods)**.** Let

$$V = \{1, 2, 3\}$$

and take $r = 2$, so

$$F(V) = \mathcal{P}_f^2(V) = \mathcal{P}_f\big(\mathcal{P}_f(V)\big).$$

Define $\gamma : V \to \mathcal{P}_f^2(V)$ by

$$\gamma(1) = \big\{\{2,3\}, \{2\}\big\}, \qquad \gamma(2) = \big\{\{1\}\big\}, \qquad \gamma(3) = \big\{\{1,2\}, \emptyset\big\}.$$

Each value $\gamma(v)$ is a finite set of finite subsets of V, hence $\gamma(v) \in \mathcal{P}_f^2(V)$. Therefore

$$\mathsf{CNNG} = (V, \gamma)$$

is a Coalgebraic Nested-Neighborhood Graph in the sense of Definition 5.14.1. Intuitively, $\gamma(1)$ assigns to vertex 1 two alternative neighbor-sets $\{2,3\}$ and $\{2\}$, while $\gamma(3)$ assigns the neighbor-set $\{1,2\}$ together with the empty neighbor-set, illustrating a 2-nested neighborhood structure. An overview diagram of this example is provided in Fig. 5.9.

Theorem 5.14.3 (Well-definedness of the r-nested CNNG construction)**.** *Fix an integer $r \geq 1$. Let $\mathcal{P}_f : \mathbf{Set} \to \mathbf{Set}$ denote the finite-powerset functor, i.e.*

$$\mathcal{P}_f(V) = \{S \subseteq V \mid S \text{ is finite}\},$$

and for a map $f : V \to W$,

$$\mathcal{P}_f(f) : \mathcal{P}_f(V) \to \mathcal{P}_f(W), \qquad \mathcal{P}_f(f)(S) = f[S] = \{f(x) \mid x \in S\}.$$

Then the r-fold iterate

$$F_r := \mathcal{P}_f^r : \mathbf{Set} \to \mathbf{Set}$$

is a well-defined endofunctor. Consequently, for every set V and every map

$$\gamma : V \to \mathcal{P}_f^r(V),$$

the pair (V, γ) is a well-defined F_r-coalgebra, hence a well-defined Coalgebraic Nested-Neighborhood Graph (CNNG) of nesting depth r.

Moreover, when $r = 1$, $\gamma(v) \in \mathcal{P}_f(V)$ is an ordinary finite neighborhood of v; when $r \geq 2$, $\gamma(v) \in \mathcal{P}_f^r(V)$ is an r-nested finite neighborhood.

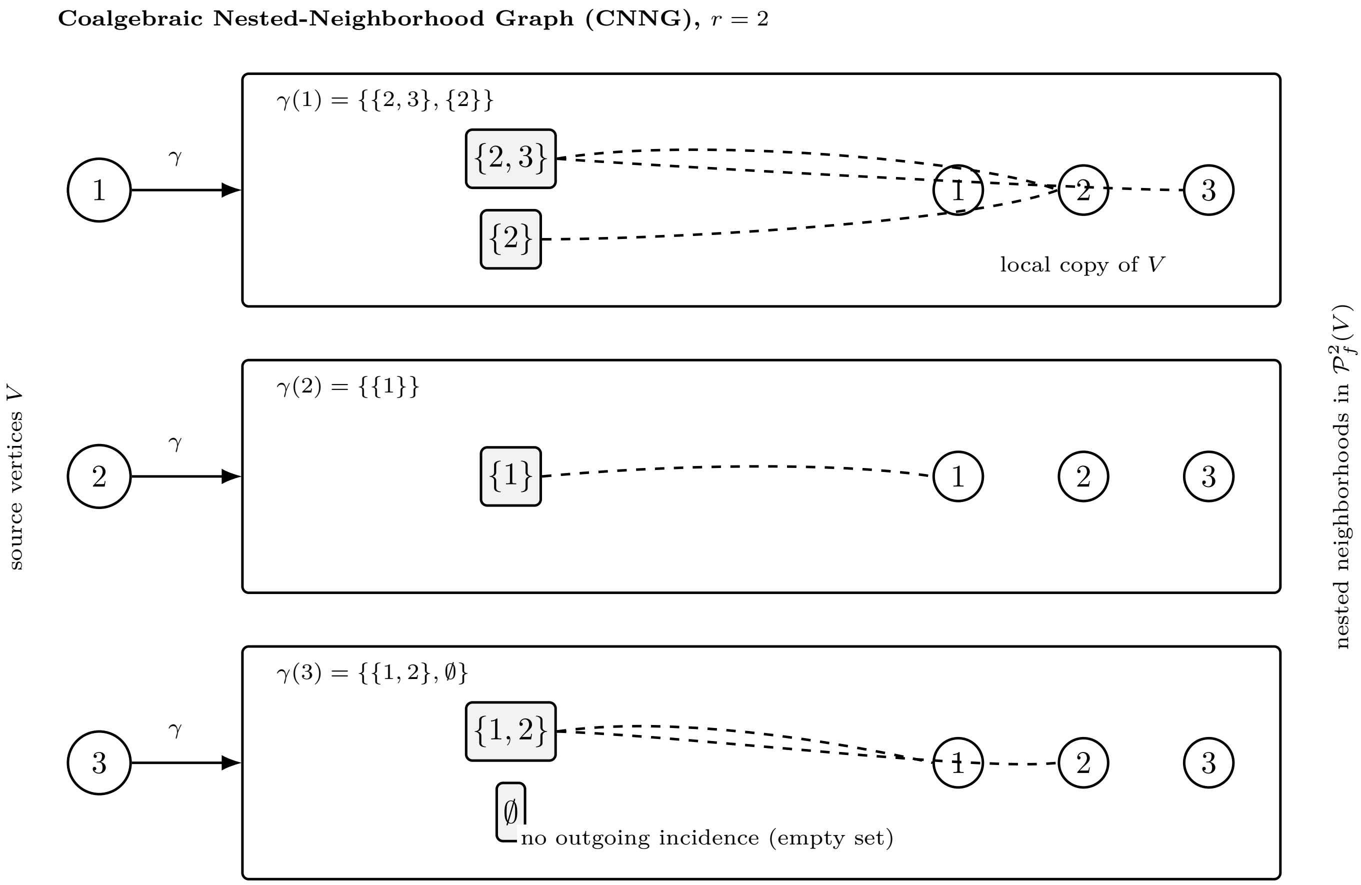


Figure 5.9: A Coalgebraic Nested-Neighborhood Graph with 2-nested neighborhoods on $V = \{1, 2, 3\}$.

Proof. We proceed in two steps.

Step 1: $\mathcal{P}_f$ is a well-defined endofunctor on Set.
For any set V, $\mathcal{P}_f(V)$ is a set (the set of finite subsets of V). For any map $f : V \to W$, define

$$\mathcal{P}_f(f)(S) = f[S] \subseteq W \qquad (S \in \mathcal{P}_f(V)).$$

This is well-defined because the image of a finite set under any map is finite; hence $f[S] \in \mathcal{P}_f(W)$.
We verify the functorial axioms.

1. *Identity:* For the identity map $\mathrm{id}_V : V \to V$,

$$\mathcal{P}_f(\mathrm{id}_V)(S) = \mathrm{id}_V[S] = S \qquad \text{for all } S \in \mathcal{P}_f(V),$$

so $\mathcal{P}_f(\mathrm{id}_V) = \mathrm{id}_{\mathcal{P}_f(V)}$.

2. *Composition:* Let $f : U \to V$ and $g : V \to W$. For any $S \in \mathcal{P}_f(U)$,

$$\mathcal{P}_f(g \circ f)(S) = (g \circ f)[S] = g[f[S]] = \mathcal{P}_f(g)\big(\mathcal{P}_f(f)(S)\big).$$

Hence

$$\mathcal{P}_f(g \circ f) = \mathcal{P}_f(g) \circ \mathcal{P}_f(f).$$

Therefore $\mathcal{P}_f$: **Set** $\to$ **Set** is an endofunctor.

Step 2: The iterate $\mathcal{P}_f^r$ is a well-defined endofunctor.
Since $\mathcal{P}_f$ is an endofunctor, its r-fold composition

$$\mathcal{P}_f^r = \underbrace{\mathcal{P}_f \circ \cdots \circ \mathcal{P}_f}_{r \text{ times}}$$

is again an endofunctor on **Set**. Explicitly:

- on objects, $V \mapsto \mathcal{P}_f^r(V)$;
- on morphisms, $f \mapsto \mathcal{P}_f^r(f)$, obtained by iterating direct image.

Functoriality (preservation of identities and composition) follows from the functoriality of $\mathcal{P}_f$ and closure of endofunctors under composition.

Now, by the definition of an F-coalgebra for an endofunctor F, any pair

$$(V, \gamma), \qquad \gamma : V \to F(V),$$

is an F-coalgebra. Taking $F = F_r = \mathcal{P}_f^r$, any map

$$\gamma : V \to \mathcal{P}_f^r(V)$$

therefore defines a well-formed coalgebra (V, γ). This is exactly a CNNG of nesting depth r.

Finally, for $r = 1$, $\gamma(v) \in \mathcal{P}_f(V)$ is just a finite subset of V, i.e. an ordinary finite neighborhood. For $r \geq 2$, $\gamma(v)$ is a finite set of $(r-1)$-nested finite subsets, so it encodes iterated (nested) neighborhood data. This proves the claim. □

5.15 Curried Graph

A curried graph has curried functions as vertices; edges are structure-preserving maps between domains and codomains satisfying commutative evaluation [265]. This is a graph concept derived from the notion of *curried functions* [266] in programming languages.

Definition 5.15.1 (Curried k-ary Graph (Curried Graph Function))**.** [265] Fix an integer $k \geq 1$. Let Σ be a finite family of *type signatures*

$$\Sigma = \{(A_1, \ldots, A_k; Z)\},$$

where each A_i and Z is a set. A *curried k-ary graph* is a directed graph

$$G^{(k)} = (V, \mathcal{E}, s, t)$$

specified as follows:

1. **(Vertices)** V is a nonempty set of curried k-ary functions of signatures in Σ, i.e.

$$V \;\subseteq \bigcup_{(A_1,\ldots,A_k;Z)\in\Sigma} \{\, f \mid f : A_1 \to (A_2 \to \cdots \to (A_k \to Z)\cdots)\,\}.$$

   When $f \in V$ has signature $(A_1, \ldots, A_k; Z)$, we write it suggestively as

$$f : \; A_1 \to A_2 \to \cdots \to A_k \to Z.$$

2. **(Edges)** An element $E \in \mathcal{E}$ is an *edge datum* between a source vertex

$$f : \; A_1 \to \cdots \to A_k \to Z$$

   and a target vertex

$$g : \; A_1' \to \cdots \to A_k' \to Z'$$

   given by a tuple of set maps

$$E = \big(E(A_1), \ldots, E(A_k), E(Z)\big),$$

   where

$$E(A_i) : A_i \to A_i' \quad (1 \leq i \leq k), \qquad E(Z) : Z \to Z'.$$

   We declare that E is a directed edge $E : f \to g$ (so $s(E) = f$ and $t(E) = g$) if and only if the following *commuting (naturality) condition* holds:

$$\forall a_1 \in A_1, \ldots, a_k \in A_k : \quad E(Z)\big(f(a_1)(a_2)\cdots(a_k)\big) = g\big(E(A_1)(a_1)\big)\big(E(A_2)(a_2)\big)\cdots\big(E(A_k)(a_k)\big).$$

Example 5.15.2 (Real-world example: migrating an e-commerce scoring function as a curried graph)**.** Consider an e-commerce platform that assigns a relevance score to each user–item pair. Let

$$A = \{\mathsf{u}_1, \mathsf{u}_2\} \quad \text{(legacy user IDs)}, \qquad B = \{\mathsf{i}_1, \mathsf{i}_2, \mathsf{i}_3\} \quad \text{(legacy item IDs)}, \qquad Z = \mathbb{R} \quad \text{(raw scores)}.$$

A legacy scoring rule can be modeled as a curried function

$$f : \; A \to B \to Z,$$

where $f(\mathsf{u})(\mathsf{i})$ is the raw relevance score of item i for user u.

Suppose the platform migrates to a new ID system and a new scoring scale:

$$A' = \{\mathsf{U}_1, \mathsf{U}_2\}, \qquad B' = \{\mathsf{I}_1, \mathsf{I}_2, \mathsf{I}_3\}, \qquad Z' = [0,1].$$

Let $g : A' \to B' \to Z'$ be the new (normalized) scoring rule. Define the migration maps

$$E(A) : A \to A', \qquad E(B) : B \to B', \qquad E(Z) : Z \to Z'$$

by

$$E(A)(\mathsf{u}_j) = \mathsf{U}_j \ (j = 1,2), \qquad E(B)(\mathsf{i}_\ell) = \mathsf{I}_\ell \ (\ell = 1,2,3),$$

and (for some fixed scale parameter $M > 0$)

$$E(Z)(x) = \min\{1, \max\{0, x/M\}\} \in [0,1].$$

Assume the new system is designed so that the normalized score equals the normalized legacy score after ID mapping, namely

$$g\big(E(A)(\mathsf{u})\big)\big(E(B)(\mathsf{i})\big) = E(Z)\big(f(\mathsf{u})(\mathsf{i})\big) \quad \text{for all } \mathsf{u} \in A, \ \mathsf{i} \in B.$$

Equivalently,

$$E(Z)\big(f(\mathsf{u})(\mathsf{i})\big) = g\big(E(A)(\mathsf{u})\big)\big(E(B)(\mathsf{i})\big) \quad \text{for all } \mathsf{u} \in A, \ \mathsf{i} \in B,$$

which is exactly the commuting (naturality) condition. Hence the triple $E = (E(A), E(B), E(Z))$ defines a directed edge $E : f \to g$ in the curried graph whose vertices are curried scoring functions, representing a consistent migration of user IDs, item IDs, and score scales.

5.16 Depth-r iterated subdivisions of polyhedral complexes

We define *depth-r iterated polyhedral complexes* (i.e., iterated subdivisions) recursively.

Definition 5.16.1 (Depth-r iterated polyhedral complex)**.** A *depth-*0 *iterated polyhedral complex* is an ordinary polyhedral complex K.

Assume that depth-r iterated polyhedral complexes have been defined. A *depth-*$(r+1)$ *iterated polyhedral complex* is a pair (K, S) such that:

1. K is a polyhedral complex.
2. S assigns to each cell $P \in K$ a polyhedral complex $S(P)$ with underlying space

   $$|S(P)| = P,$$

   and the assignment is *face-compatible* in the sense that for every face $F \leq P$ (hence $F \in K$),

   $$S(F) = \{\, Q \in S(P) \mid Q \subseteq F \,\} = S(P) \cap F.$$

3. Moreover, for each $P \in K$, the complex $S(P)$ is equipped with the structure of a *depth-r iterated polyhedral complex*, and the above face-compatibility condition holds at *every depth level* (i.e., the induced iterated subdivision on each face agrees with the iterated subdivision assigned to that face).

Remark 5.16.2 (Refinement chains (flags))**.** *Equivalently, a depth-r iterated polyhedral complex can be described by specifying a chain of refinements (a* flag*)*

$$K^{(0)} \prec K^{(1)} \prec \cdots \prec K^{(r)},$$

where $K^{(0)}$ is the initial polyhedral complex and, for each $\ell = 0, \ldots, r-1$, the complex $K^{(\ell+1)}$ is a coherent (face-compatible) polyhedral subdivision of $K^{(\ell)}$.

Example 5.16.3 (A depth-2 iterated polyhedral complex on the unit square)**.** Let K be the polyhedral complex in $\mathbb{R}^2$ whose underlying space is the unit square

$$|K| = [0,1] \times [0,1],$$

obtained by subdividing the square into two triangles:

$$P_1 = \mathrm{conv}\{(0,0),(1,0),(0,1)\}, \qquad P_2 = \mathrm{conv}\{(1,0),(1,1),(0,1)\},$$

together with all faces of P_1 and P_2 (edges and vertices). Thus K is a depth-0 iterated polyhedral complex.

First refinement (depth 1). Define S_1 by assigning to each 2-cell P_i the barycentric subdivision into four triangles. For example, for P_1 let m_1 be the midpoint of the edge from $(1,0)$ to $(0,1)$, and set

$$S_1(P_1) = \Big\{\mathrm{conv}\{(0,0),(1,0),m_1\},\ \mathrm{conv}\{(0,0),(0,1),m_1\},$$

$$\mathrm{conv}\{(0,0),m_1,\tfrac{1}{2}(1,0)\},\ \mathrm{conv}\{(0,0),m_1,\tfrac{1}{2}(0,1)\}\Big\},$$

together with all faces (the precise choice of four triangles is not essential; any polyhedral subdivision of P_1 works). Define $S_1(P_2)$ analogously. For each edge (a 1-face) $F \leq P_i$, define $S_1(F)$ to be the induced subdivision of F by the endpoints and any new vertices introduced on that edge; for each vertex v, set $S_1(\{v\}) = \{\{v\}\}$. Then S_1 is face-compatible, and (K, S_1) is a depth-1 iterated polyhedral complex.

Second refinement (depth 2). Define S_2 by further subdividing each triangle $Q \in S_1(P_i)$ into two triangles by drawing a median, and extending this assignment to faces by restriction. Concretely, for each 2-cell $Q \in S_1(P_i)$ choose one edge, insert its midpoint (if not already present), and split Q into two smaller triangles; include all faces. For a face $F \leq Q$, define $S_2(F) = S_2(Q) \cap F$.

Now set, for each original cell $P \in K$,

$$S(P) := \big(S_1(P), S_2|_{S_1(P)}\big),$$

i.e., $S(P)$ is equipped with the depth-1 iterated structure given by the second refinement inside P. By construction, $|S(P)| = P$ and the face-compatibility condition holds at both refinement levels. Hence (K, S) is a depth-2 iterated polyhedral complex in the sense of Definition 5.16.1.

5.17 Sheaf HyperGraph / Sheaf SuperHyperGraph

A Sheaf HyperGraph assigns local data spaces to vertices and hyperedges, with restriction maps along incidences, enabling coherent local-to-global information. A Sheaf SuperHyperGraph assigns local data spaces to supervertices and superhyperedges, with restriction maps encoding coherent hierarchical local-to-global information structures.

Definition 5.17.1 (Incidence poset of a HyperGraph)**.** Let $H = (V, E)$ be a HyperGraph. Its *incidence poset* is the poset

$$\mathrm{Inc}(H) := (V \sqcup E, \preceq),$$

where the order relation is generated by

$$v \preceq e \quad \Longleftrightarrow \quad v \in e \qquad (v \in V,\ e \in E).$$

Thus vertices are below the hyperedges that contain them.

Definition 5.17.2 (Sheaf HyperGraph)**.** Fix a field $\mathbb{K}$. A *Sheaf HyperGraph* is a pair

$$(H, \mathcal{F}),$$

where $H = (V, E)$ is a HyperGraph and

$$\mathcal{F} : \mathrm{Inc}(H)^{\mathrm{op}} \to \mathbf{Vect}_{\mathbb{K}}$$

is a contravariant functor.

Equivalently, a Sheaf HyperGraph consists of:

1. a $\mathbb{K}$-vector space $\mathcal{F}(x)$ for each $x \in V \sqcup E$;
2. for every incidence $v \in e$, a linear *restriction map*

$$\rho_{e,v} : \mathcal{F}(e) \to \mathcal{F}(v);$$

3. identity and functoriality conditions inherited from the poset structure.

The spaces $\mathcal{F}(v)$ and $\mathcal{F}(e)$ are called the *stalks* at the vertex v and the hyperedge e, respectively.

Definition 5.17.3 (Incidence poset of an n-SuperHyperGraph)**.** Let $\mathcal{H}^{(n)} = (V, E)$ be an n-SuperHyperGraph. Its *superincidence poset* is the poset

$$\mathrm{SInc}\big(\mathcal{H}^{(n)}\big) := (V \sqcup E, \preceq),$$

where

$$v \preceq e \iff v \in e \qquad (v \in V,\ e \in E).$$

Definition 5.17.4 (Sheaf n-SuperHyperGraph)**.** Fix a field $\mathbb{K}$. A *Sheaf n-SuperHyperGraph* is a pair

$$\big(\mathcal{H}^{(n)}, \mathcal{F}\big),$$

where $\mathcal{H}^{(n)} = (V, E)$ is an n-SuperHyperGraph and

$$\mathcal{F} : \mathrm{SInc}\big(\mathcal{H}^{(n)}\big)^{\mathrm{op}} \to \mathbf{Vect}_{\mathbb{K}}$$

is a contravariant functor.

Equivalently, it assigns:

1. a $\mathbb{K}$-vector space $\mathcal{F}(x)$ to each supervertex or superhyperedge $x \in V \sqcup E$;
2. for every superincidence $v \in e$, a restriction map

$$\rho_{e,v} : \mathcal{F}(e) \to \mathcal{F}(v).$$

Remark 5.17.5. *When $n = 0$, a Sheaf 0-SuperHyperGraph is precisely a Sheaf HyperGraph.*

Example 5.17.6 (A Sheaf 1-SuperHyperGraph)**.** Let the base set be

$$V_0 = \{a, b, c\},$$

and take $n = 1$, so that

$$\mathcal{P}^1(V_0) = \mathcal{P}(V_0).$$

Fix the field

$$\mathbb{K} = \mathbb{R}.$$

Step 1: The underlying 1-SuperHyperGraph. Define the following 1-supervertices:

$$v_1 := \{a, b\}, \qquad v_2 := \{c\}, \qquad v_3 := \{a, c\}.$$

Set

$$V = \{v_1, v_2, v_3\} \subseteq \mathcal{P}(V_0).$$

Define the superhyperedge family by

$$E = \{e_1, e_2\}, \qquad e_1 = \{v_1, v_2\}, \qquad e_2 = \{v_1, v_3\}.$$

Hence

$$\mathcal{H}^{(1)} = (V, E)$$

is a finite 1-SuperHyperGraph.

Step 2: The stalks of the sheaf. Define a vector space for each supervertex and each superhyperedge by

$$\mathcal{F}(v_1) = \mathbb{R}^2, \qquad \mathcal{F}(v_2) = \mathbb{R}, \qquad \mathcal{F}(v_3) = \mathbb{R}^2,$$

and

$$\mathcal{F}(e_1) = \mathbb{R}^2, \qquad \mathcal{F}(e_2) = \mathbb{R}^3.$$

Step 3: Restriction maps along superincidences. Since

$$v_1 \in e_1, \quad v_2 \in e_1, \quad v_1 \in e_2, \quad v_3 \in e_2,$$

we define the restriction maps

$$\rho_{e_1,v_1} : \mathbb{R}^2 \to \mathbb{R}^2, \qquad \rho_{e_1,v_1}(x,y) = (x,y),$$
$$\rho_{e_1,v_2} : \mathbb{R}^2 \to \mathbb{R}, \qquad \rho_{e_1,v_2}(x,y) = x+y,$$
$$\rho_{e_2,v_1} : \mathbb{R}^3 \to \mathbb{R}^2, \qquad \rho_{e_2,v_1}(x,y,z) = (x,y),$$

and

$$\rho_{e_2,v_3} : \mathbb{R}^3 \to \mathbb{R}^2, \qquad \rho_{e_2,v_3}(x,y,z) = (y,z).$$

For every object $x \in V \sqcup E$, let

$$\rho_{x,x} = \mathrm{id}_{\mathcal{F}(x)}.$$

Step 4: The associated contravariant functor. The superincidence poset $\mathrm{SInc}(\mathcal{H}^{(1)})$ has objects

$$V \sqcup E = \{v_1, v_2, v_3, e_1, e_2\},$$

and its only non-identity order relations are

$$v_1 \preceq e_1, \qquad v_2 \preceq e_1, \qquad v_1 \preceq e_2, \qquad v_3 \preceq e_2.$$

Passing to the opposite category reverses these arrows, so the above restriction maps define a contravariant functor

$$\mathcal{F} : \mathrm{SInc}\big(\mathcal{H}^{(1)}\big)^{\mathrm{op}} \to \mathbf{Vect}_{\mathbb{R}}.$$

Therefore,

$$\big(\mathcal{H}^{(1)}, \mathcal{F}\big)$$

is a Sheaf 1-SuperHyperGraph.

The vector spaces attached to the supervertices represent local data spaces on grouped entities, while the vector spaces attached to the superhyperedges represent joint data spaces on higher-order interactions. The restriction maps extract or aggregate local information from each superhyperedge to its incident supervertices.

Theorem 5.17.7 (Well-definedness of Sheaf n-SuperHyperGraphs)**.** *Let $\mathbb{K}$ be a field, let V_0 be a finite nonempty set, let $n \in \mathbb{N}_0$, and let*

$$\mathcal{H}^{(n)} = (V, E)$$

be an n-SuperHyperGraph. Assume that its superincidence poset is

$$\mathrm{SInc}\big(\mathcal{H}^{(n)}\big) := (V \sqcup E, \preceq),$$

where $\preceq$ is the least reflexive relation satisfying

$$v \preceq e \iff v \in e \qquad (v \in V,\ e \in E).$$

Then the following hold:

1. $\mathrm{SInc}\big(\mathcal{H}^{(n)}\big)$ *is a well-defined finite poset;*
2. *its opposite category*
$$\mathrm{SInc}\big(\mathcal{H}^{(n)}\big)^{\mathrm{op}}$$
is a well-defined small category;
3. *consequently, every contravariant functor*
$$\mathcal{F} : \mathrm{SInc}\big(\mathcal{H}^{(n)}\big)^{\mathrm{op}} \to \mathbf{Vect}_{\mathbb{K}}$$
is a well-defined mathematical object, and hence the pair
$$\big(\mathcal{H}^{(n)}, \mathcal{F}\big)$$
is a well-defined Sheaf n-SuperHyperGraph.

Proof. Since $\mathcal{H}^{(n)} = (V, E)$ is an n-SuperHyperGraph, the set V is finite and satisfies

$$V \subseteq \mathcal{P}^n(V_0),$$

while E is a finite family of nonempty subsets of V. Hence the disjoint union

$$V \sqcup E$$

is a finite set.

We first verify that $\preceq$ defines a partial order on $V \sqcup E$.

Reflexivity. By construction, $\preceq$ is reflexive.

Antisymmetry. Suppose $x, y \in V \sqcup E$ satisfy $x \preceq y$ and $y \preceq x$. If either relation is non-identity, then by definition one element must be a supervertex and the other a superhyperedge. More precisely, a nontrivial relation has the form

$$v \preceq e \qquad (v \in V,\ e \in E,\ v \in e).$$

There is no nontrivial relation of the form $e \preceq v$, because the generating incidence relation only goes from supervertices to superhyperedges. Therefore $x \preceq y$ and $y \preceq x$ can simultaneously hold only when $x = y$. Thus $\preceq$ is antisymmetric.

Transitivity. Let $x, y, z \in V \sqcup E$ with $x \preceq y$ and $y \preceq z$. If either relation is an identity, transitivity is immediate. So it remains to consider non-identity relations. But every non-identity relation has the form

$$v \preceq e \qquad (v \in V,\ e \in E).$$

Hence there is no chain of two distinct non-identity arrows: a superhyperedge cannot be below any distinct element, and no distinct element can lie below a supervertex. Therefore every composable pair reduces to a case involving an identity morphism, and transitivity follows.

Thus $(V \sqcup E, \preceq)$ is a well-defined finite poset. This proves (1).

Every poset canonically determines a small category: objects are the elements of the poset, and there is a unique morphism

$$x \to y \quad \iff \quad x \preceq y.$$

Since $V \sqcup E$ is finite, this category is small. Therefore its opposite category

$$\mathrm{SInc}\big(\mathcal{H}^{(n)}\big)^{\mathrm{op}}$$

is also well-defined and small. This proves (2).

Finally, $\mathbf{Vect}_{\mathbb{K}}$ is a well-defined category because $\mathbb{K}$ is a field. Hence any contravariant functor

$$\mathcal{F} : \mathrm{SInc}\big(\mathcal{H}^{(n)}\big)^{\mathrm{op}} \to \mathbf{Vect}_{\mathbb{K}}$$

is a well-defined functor from a small category to $\mathbf{Vect}_{\mathbb{K}}$. Therefore the pair

$$\big(\mathcal{H}^{(n)}, \mathcal{F}\big)$$

is well-defined as a Sheaf n-SuperHyperGraph. This proves (3). □

Proposition 5.17.8 (Equivalent local description)**.** *Let $\mathcal{H}^{(n)} = (V, E)$ be an n-SuperHyperGraph. Giving a contravariant functor*

$$\mathcal{F} : \mathrm{SInc}\big(\mathcal{H}^{(n)}\big)^{\mathrm{op}} \to \mathbf{Vect}_{\mathbb{K}}$$

is equivalent to giving:

1. *a $\mathbb{K}$-vector space $\mathcal{F}(x)$ for each $x \in V \sqcup E$;*
2. *for each superincidence $v \in e$, a linear map*

$$\rho_{e,v} : \mathcal{F}(e) \to \mathcal{F}(v);$$

3. *the identity maps*

$$\rho_{x,x} = \mathrm{id}_{\mathcal{F}(x)} \qquad (x \in V \sqcup E).$$

No additional cocycle condition is required beyond identities, because in $\mathrm{SInc}(\mathcal{H}^{(n)})$ *there are no nontrivial composable chains of length* 2.

Proof. A functor on a poset category assigns an object to each element and a morphism to each order relation, preserving identities and composition. Here the only non-identity order relations are the incidences

$$v \preceq e \iff v \in e.$$

Passing to the opposite category reverses these arrows, so each incidence gives a unique morphism

$$e \to v,$$

which is exactly the restriction map

$$\rho_{e,v} : \mathcal{F}(e) \to \mathcal{F}(v).$$

Since there are no nontrivial composable chains of two distinct incidences, functoriality imposes no further compatibility except preservation of identities. Hence the two descriptions are equivalent. □

5.18 Fibered HyperGraph / Fibered SuperHyperGraph

A Fibered HyperGraph assigns each vertex a local state space and each hyperedge a compatibility relation on those fibers collectively. A Fibered SuperHyperGraph assigns each supervertex a local state space and each superhyperedge a compatibility relation on those fibers collectively.

Definition 5.18.1 (Fibered HyperGraph)**.** A *Fibered HyperGraph* is a quadruple

$$\mathfrak{F} = (V, E, \{F_v\}_{v\in V}, \{R_e\}_{e\in E}),$$

such that:

1. $H = (V, E)$ is a finite HyperGraph;
2. for each vertex $v \in V$, F_v is a nonempty set, called the *fiber over* v;
3. for each hyperedge $e \in E$, one is given a relation

$$R_e \subseteq \prod_{v\in e} F_v.$$

Remark 5.18.2. *The relation* R_e *describes the admissible joint states of the fibers attached to the vertices belonging to the hyperedge* e.

Definition 5.18.3 (Total space of a Fibered HyperGraph)**.** Let

$$\mathfrak{F} = (V, E, \{F_v\}_{v\in V}, \{R_e\}_{e\in E})$$

be a Fibered HyperGraph. Its *total space* is the disjoint union

$$X := \bigsqcup_{v\in V} F_v,$$

equipped with the natural projection

$$\pi : X \to V, \qquad \pi(x) = v \text{ if } x \in F_v.$$

Definition 5.18.4 (Fibered n-SuperHyperGraph)**.** Let $\mathcal{H}^{(n)} = (V, E)$ be a finite n-SuperHyperGraph. A *Fibered* n*-SuperHyperGraph* is a quadruple

$$\mathfrak{F}^{(n)} = (V, E, \{F_v\}_{v\in V}, \{R_e\}_{e\in E}),$$

such that:

1. $\mathcal{H}^{(n)} = (V, E)$ is an n-SuperHyperGraph;
2. for each supervertex $v \in V$, F_v is a nonempty set;

3. for each superhyperedge $e \in E$, one is given a relation
$$R_e \subseteq \prod_{v \in e} F_v.$$

Remark 5.18.5. *When $n = 0$, a Fibered 0-SuperHyperGraph is exactly a Fibered HyperGraph.*

Example 5.18.6 (A Fibered 1-SuperHyperGraph for team-state compatibility)**.** Let the base set be
$$V_0 = \{a, b, c, d\},$$
and take $n = 1$, so that
$$\mathcal{P}^1(V_0) = \mathcal{P}(V_0).$$

Define the following 1-supervertices:
$$v_1 := \{a, b\}, \qquad v_2 := \{c\}, \qquad v_3 := \{d\}.$$

Set
$$V = \{v_1, v_2, v_3\} \subseteq \mathcal{P}(V_0).$$

Define the superhyperedge family by
$$E = \{e_1, e_2\}, \qquad e_1 = \{v_1, v_2\}, \qquad e_2 = \{v_1, v_3\}.$$

Then
$$\mathcal{H}^{(1)} = (V, E)$$
is a finite 1-SuperHyperGraph.

Next, assign a nonempty fiber to each supervertex:
$$F_{v_1} = \{\text{low}, \text{high}\}, \qquad F_{v_2} = \{\text{off}, \text{on}\}, \qquad F_{v_3} = \{\text{idle}, \text{busy}\}.$$

Here one may interpret:

- F_{v_1} as the activity level of the grouped team $\{a, b\}$,
- F_{v_2} as the availability state of the singleton team $\{c\}$,
- F_{v_3} as the workload state of the singleton team $\{d\}$.

Now define a relation on each superhyperedge.
For
$$e_1 = \{v_1, v_2\},$$
the corresponding product is
$$\prod_{v \in e_1} F_v = F_{v_1} \times F_{v_2},$$
and define
$$R_{e_1} = \{(\text{low}, \text{off}),\ (\text{high}, \text{on})\} \subseteq F_{v_1} \times F_{v_2}.$$

For
$$e_2 = \{v_1, v_3\},$$
the corresponding product is
$$\prod_{v \in e_2} F_v = F_{v_1} \times F_{v_3},$$
and define
$$R_{e_2} = \{(\text{low}, \text{idle}),\ (\text{high}, \text{busy})\} \subseteq F_{v_1} \times F_{v_3}.$$

Therefore,
$$\mathfrak{F}^{(1)} = \big(V, E, \{F_v\}_{v \in V}, \{R_e\}_{e \in E}\big)$$
is a Fibered 1-SuperHyperGraph.

The supervertex $v_1 = \{a, b\}$ has its own local state space, and each superhyperedge imposes a compatibility relation between the states of the participating supervertices. Thus the underlying 1-SuperHyperGraph is enriched by fiber data and edgewise state constraints.

Theorem 5.18.7 (Well-definedness of Fibered n-SuperHyperGraphs)**.** *Let V_0 be a finite nonempty set, let $n \in \mathbb{N}_0$, and let*

$$\mathcal{H}^{(n)} = (V, E)$$

be a finite n-SuperHyperGraph. Assume that:

1. *for each $v \in V$, a nonempty set F_v is given;*
2. *for each $e \in E$, a subset*

   $$R_e \subseteq \prod_{v \in e} F_v$$

   is given.

Then the quadruple

$$\mathfrak{F}^{(n)} = (V, E, \{F_v\}_{v \in V}, \{R_e\}_{e \in E})$$

is a well-defined Fibered n-SuperHyperGraph.

In particular:

1. *for every superhyperedge $e \in E$, the Cartesian product*

   $$\prod_{v \in e} F_v$$

   is well-defined;
2. *each R_e is a well-defined relation on the family of fibers indexed by the supervertices contained in e.*

Proof. Since $\mathcal{H}^{(n)} = (V, E)$ is a finite n-SuperHyperGraph, by definition we have

$$V \subseteq \mathcal{P}^n(V_0), \qquad E \subseteq \mathcal{P}(V) \setminus \{\varnothing\}.$$

Hence V is a finite set of n-supervertices, and each $e \in E$ is a nonempty finite subset of V.

Now fix $e \in E$. Because $e \subseteq V$ and V is finite, the index set e is finite. By assumption, for every $v \in e$, the fiber F_v is a nonempty set. Therefore the indexed Cartesian product

$$\prod_{v \in e} F_v$$

is well-defined as the set of all tuples

$$(x_v)_{v \in e} \quad \text{such that} \quad x_v \in F_v \text{ for every } v \in e.$$

Since e is finite and each $F_v \neq \varnothing$, this product is a well-defined set; moreover, it is nonempty.

Again by assumption, for each $e \in E$ one is given a subset

$$R_e \subseteq \prod_{v \in e} F_v.$$

Hence R_e is a well-defined relation among the fibers corresponding to the supervertices in the superhyperedge e.

Since this construction is valid for every $e \in E$, all components of

$$\mathfrak{F}^{(n)} = (V, E, \{F_v\}_{v \in V}, \{R_e\}_{e \in E})$$

are well-defined and satisfy the defining conditions of a Fibered n-SuperHyperGraph. Therefore $\mathfrak{F}^{(n)}$ is well-defined. □

5.19 Galois HyperGraph / Galois SuperHyperGraph

A Galois HyperGraph is a hypergraph whose hyperedges are nonempty Galois-closed vertex sets induced by a formal context. A Galois SuperHyperGraph is a superhypergraph whose superhyperedges are nonempty Galois-closed supervertex sets induced by a formal context.

Definition 5.19.1 (Formal context)**.** A *formal context* is a triple

$$\mathbb{C} = (X, M, \mathcal{I}),$$

where X is a nonempty set of objects, M is a nonempty set of attributes, and

$$\mathcal{I} \subseteq X \times M$$

is an incidence relation.

For $A \subseteq X$ and $B \subseteq M$, define the derivation operators

$$A^{\uparrow} := \{\, m \in M \mid \forall x \in A,\ (x, m) \in \mathcal{I} \,\},$$

$$B^{\downarrow} := \{\, x \in X \mid \forall m \in B,\ (x, m) \in \mathcal{I} \,\}.$$

Then the operator

$$\mathrm{cl}(A) := A^{\uparrow\downarrow}$$

is the associated Galois closure on subsets of X.

Definition 5.19.2 (Galois HyperGraph)**.** A *Galois HyperGraph* is a quadruple

$$\mathfrak{G} = (V, M, \mathcal{I}, E),$$

such that:

1. $(V, M, \mathcal{I})$ is a finite formal context;
2. $$E \subseteq \{\, A \subseteq V \mid A \neq \varnothing,\ A^{\uparrow\downarrow} = A \,\}.$$

The hyperedges in E are called *Galois hyperedges*; they are precisely the chosen nonempty closed subsets of the object set V under the Galois closure operator.

Remark 5.19.3. *Thus a Galois HyperGraph is a HyperGraph whose admissible hyperedges are constrained by closure in a formal context.*

Definition 5.19.4 (Galois n-SuperHyperGraph)**.** Let V_0 be a finite nonempty base set and let $n \in \mathbb{N}_0$. A *Galois n-SuperHyperGraph* is a quadruple

$$\mathfrak{G}^{(n)} = (V, M, \mathcal{I}, E),$$

such that:

1. $$V \subseteq \mathcal{P}^n(V_0)$$
   is a finite set of n-supervertices;
2. $(V, M, \mathcal{I})$ is a formal context;
3. $$E \subseteq \{\, A \subseteq V \mid A \neq \varnothing,\ A^{\uparrow\downarrow} = A \,\}.$$

The members of E are called *Galois superhyperedges.*

Remark 5.19.5. *When $n = 0$, a Galois 0-SuperHyperGraph reduces to a Galois HyperGraph.*

Example 5.19.6 (A Galois 1-SuperHyperGraph)**.** Let the finite nonempty base set be

$$V_0 = \{a, b, c\},$$

and take $n = 1$. Then

$$\mathcal{P}^1(V_0) = \mathcal{P}(V_0).$$

We choose the following 1-supervertices:

$$v_1 := \{a\}, \qquad v_2 := \{b\}, \qquad v_3 := \{a, b\}.$$

Set

$$V = \{v_1, v_2, v_3\} \subseteq \mathcal{P}(V_0).$$

Next, let the attribute set be

$$M = \{m_1, m_2\}.$$

Interpret m_1 as a common basic property and m_2 as an additional higher-level property. Define the incidence relation

$$\mathcal{I} \subseteq V \times M$$

by

$$\mathcal{I} = \{(v_1, m_1),\ (v_2, m_1),\ (v_3, m_1),\ (v_3, m_2)\}.$$

Thus the incidences are:

$$v_1^{\uparrow} = \{m_1\}, \qquad v_2^{\uparrow} = \{m_1\}, \qquad v_3^{\uparrow} = \{m_1, m_2\}.$$

We now compute some Galois closures.

(i) The singleton $\{v_3\}$**.** We have

$$\{v_3\}^{\uparrow} = \{m_1, m_2\}.$$

Hence

$$\{v_3\}^{\uparrow\downarrow} = \{\, v \in V \mid (v, m_1) \in \mathcal{I} \text{ and } (v, m_2) \in \mathcal{I} \,\} = \{v_3\}.$$

Therefore $\{v_3\}$ is $\mathcal{I}$-closed.

(ii) The full set $V = \{v_1, v_2, v_3\}$**.** Since m_1 is the only attribute common to all three supervertices,

$$V^{\uparrow} = \{m_1\}.$$

Thus

$$V^{\uparrow\downarrow} = \{\, v \in V \mid (v, m_1) \in \mathcal{I} \,\} = V.$$

Hence V is also $\mathcal{I}$-closed.

Now define

$$E = \big\{\{v_3\},\ \{v_1, v_2, v_3\}\big\}.$$

Each member of E is nonempty and Galois-closed, so

$$E \subseteq \{\, A \subseteq V \mid A \neq \varnothing,\ A^{\uparrow\downarrow} = A \,\}.$$

Therefore

$$\mathfrak{G}^{(1)} = (V, M, \mathcal{I}, E)$$

is a Galois 1-SuperHyperGraph.

The supervertex $v_3 = \{a, b\}$ forms a closed singleton because it alone possesses the additional attribute m_2. Meanwhile, the whole family V forms another closed set because all supervertices share the common attribute m_1. Thus the Galois superhyperedges encode closure-stable families of supervertices determined by the formal context.

Theorem 5.19.7 (Well-definedness of Galois n-SuperHyperGraphs)**.** *Let V_0 be a finite nonempty base set, let $n \in \mathbb{N}_0$, and let*

$$V \subseteq \mathcal{P}^n(V_0)$$

be a finite set. Let M be a set and let

$$\mathcal{I} \subseteq V \times M$$

be a binary relation. Define, for every $A \subseteq V$ and $B \subseteq M$,

$$A^{\uparrow} := \{\, m \in M \mid \forall v \in A,\ (v, m) \in \mathcal{I} \,\}, \qquad B^{\downarrow} := \{\, v \in V \mid \forall m \in B,\ (v, m) \in \mathcal{I} \,\}.$$

Then the following hold:

1. *for every $A \subseteq V$, the set*
$$A^{\uparrow\downarrow} := \left(A^{\uparrow}\right)^{\downarrow}$$
is a well-defined subset of V;

2. *the family*
$$\mathrm{Cl}_{\mathcal{I}}(V) := \{\, A \subseteq V \mid A^{\uparrow\downarrow} = A \,\}$$
of $\mathcal{I}$-closed subsets of V is well-defined;

3. *therefore the family*
$$\mathrm{Cl}^{*}_{\mathcal{I}}(V) := \{\, A \subseteq V \mid A \neq \varnothing,\ A^{\uparrow\downarrow} = A \,\}$$
of nonempty $\mathcal{I}$-closed subsets of V is well-defined.

Consequently, for every choice
$$E \subseteq \mathrm{Cl}^{*}_{\mathcal{I}}(V),$$
the quadruple
$$\mathfrak{G}^{(n)} = (V, M, \mathcal{I}, E)$$
is a well-defined Galois n-SuperHyperGraph.

Proof. Since V_0 is a finite nonempty set and $n \in \mathbb{N}_0$, the iterated powerset
$$\mathcal{P}^{n}(V_0)$$
is well-defined by finite recursion. Hence the condition
$$V \subseteq \mathcal{P}^{n}(V_0)$$
is meaningful, and V is a well-defined finite set of n-supervertices.

Next, because $\mathcal{I} \subseteq V \times M$, for every subset $A \subseteq V$ the set
$$A^{\uparrow} = \{\, m \in M \mid \forall v \in A,\ (v, m) \in \mathcal{I} \,\}$$
is a well-defined subset of M. Indeed, membership of an element $m \in M$ in $A^{\uparrow}$ is determined by the precise first-order condition
$$\forall v \in A,\ (v, m) \in \mathcal{I}.$$

Similarly, for every subset $B \subseteq M$, the set
$$B^{\downarrow} = \{\, v \in V \mid \forall m \in B,\ (v, m) \in \mathcal{I} \,\}$$
is a well-defined subset of V. Therefore, for every $A \subseteq V$, the composite set
$$A^{\uparrow\downarrow} = \left(A^{\uparrow}\right)^{\downarrow}$$
is well-defined and satisfies
$$A^{\uparrow\downarrow} \subseteq V.$$
This proves (1).

Now consider the family
$$\mathrm{Cl}_{\mathcal{I}}(V) = \{\, A \subseteq V \mid A^{\uparrow\downarrow} = A \,\}.$$
Since $\mathcal{P}(V)$ is well-defined and each expression $A^{\uparrow\downarrow}$ is well-defined by (1), the condition
$$A^{\uparrow\downarrow} = A$$
is meaningful for every $A \subseteq V$. Hence $\mathrm{Cl}_{\mathcal{I}}(V)$ is a well-defined subfamily of $\mathcal{P}(V)$. This proves (2).

Removing the empty set yields
$$\mathrm{Cl}^{*}_{\mathcal{I}}(V) = \{\, A \subseteq V \mid A \neq \varnothing,\ A^{\uparrow\downarrow} = A \,\},$$
which is likewise a well-defined family of subsets of V. This proves (3).

Finally, if
$$E \subseteq \mathrm{Cl}^{*}_{\mathcal{I}}(V),$$
then every member of E is a nonempty subset of V satisfying the Galois-closure condition
$$A^{\uparrow\downarrow} = A.$$
Thus all components in the definition of a Galois n-SuperHyperGraph are well-defined, and therefore
$$\mathfrak{G}^{(n)} = (V, M, \mathcal{I}, E)$$
is a well-defined Galois n-SuperHyperGraph. □

5.20 Rewrite HyperGraph / Rewrite SuperHyperGraph

A Rewrite HyperGraph is a hypergraph equipped with rewrite rules that replace matched subhypergraphs while preserving specified interfaces. A Rewrite SuperHyperGraph is a superhypergraph equipped with rewrite rules that replace matched subsuperhypergraphs while preserving specified interfaces.

Definition 5.20.1 (HyperGraph monomorphism)**.** Let $H = (V, E)$ and $H' = (V', E')$ be HyperGraphs. A *HyperGraph monomorphism*

$$f : H \hookrightarrow H'$$

is an injective map

$$f_V : V \to V'$$

such that for every hyperedge $e \in E$, its image

$$f_V(e) := \{ f_V(v) \mid v \in e \}$$

belongs to E'.

Definition 5.20.2 (Rewrite rule for HyperGraphs)**.** A *rewrite rule for HyperGraphs* is a span

$$\rho = \big(L \xleftarrow{l} K \xrightarrow{r} R\big),$$

where L, K, R are finite HyperGraphs and

$$l : K \hookrightarrow L, \qquad r : K \hookrightarrow R$$

are HyperGraph monomorphisms.

The HyperGraph K is called the *interface* of the rule, L the *left-hand side*, and R the *right-hand side*.

Definition 5.20.3 (Rewrite HyperGraph)**.** A *Rewrite HyperGraph* is a pair

$$\mathfrak{H} = (H, \mathscr{R}),$$

where H is a finite HyperGraph and $\mathscr{R}$ is a finite set of rewrite rules for HyperGraphs.

Intuitively, each rule

$$L \leftarrow K \to R$$

specifies that a copy of L occurring inside H may be replaced by a copy of R, while preserving the common interface K.

Definition 5.20.4 (n-SuperHyperGraph monomorphism)**.** Let $\mathcal{H}^{(n)} = (V, E)$ and $\mathcal{G}^{(n)} = (V', E')$ be n-SuperHyperGraphs over the same base level n. A *monomorphism of n-SuperHyperGraphs*

$$f : \mathcal{H}^{(n)} \hookrightarrow \mathcal{G}^{(n)}$$

is an injective map

$$f_V : V \to V'$$

such that for every superhyperedge $e \in E$,

$$f_V(e) := \{ f_V(v) \mid v \in e \} \in E'.$$

Definition 5.20.5 (Rewrite rule for n-SuperHyperGraphs)**.** A *rewrite rule for n-SuperHyperGraphs* is a span

$$\rho = \big(L^{(n)} \xleftarrow{l} K^{(n)} \xrightarrow{r} R^{(n)}\big),$$

where $L^{(n)}, K^{(n)}, R^{(n)}$ are finite n-SuperHyperGraphs and

$$l : K^{(n)} \hookrightarrow L^{(n)}, \qquad r : K^{(n)} \hookrightarrow R^{(n)}$$

are monomorphisms of n-SuperHyperGraphs.

Definition 5.20.6 (Rewrite n-SuperHyperGraph)**.** A *Rewrite n-SuperHyperGraph* is a pair

$$\mathfrak{H}^{(n)} = \big(\mathcal{H}^{(n)}, \mathscr{R}^{(n)}\big),$$

where $\mathcal{H}^{(n)}$ is a finite n-SuperHyperGraph and $\mathscr{R}^{(n)}$ is a finite set of rewrite rules for n-SuperHyperGraphs.

Remark 5.20.7. *When $n = 0$, a Rewrite 0-SuperHyperGraph reduces to a Rewrite HyperGraph.*

Example 5.20.8 (A Rewrite 1-SuperHyperGraph for team reconfiguration)**.** Let the base set be

$$V_0 = \{a, b, c, d\},$$

and take $n = 1$, so that 1-supervertices are subsets of V_0.

Step 1: The underlying 1-SuperHyperGraph. Define the following 1-supervertices:

$$v_{ab} := \{a, b\}, \qquad v_c := \{c\}, \qquad v_d := \{d\}, \qquad v_{cd} := \{c, d\}.$$

Let

$$V = \{v_{ab}, v_c, v_d, v_{cd}\} \subseteq \mathcal{P}(V_0).$$

Define the superhyperedge set by

$$E = \{e_1, e_2\}, \qquad e_1 = \{v_{ab}, v_c\}, \qquad e_2 = \{v_{ab}, v_d\}.$$

Then

$$\mathcal{H}^{(1)} = (V, E)$$

is a finite 1-SuperHyperGraph.

Step 2: A rewrite rule. We now define a rewrite rule

$$\rho = \left(L^{(1)} \xleftarrow{l} K^{(1)} \xrightarrow{r} R^{(1)}\right)$$

for 1-SuperHyperGraphs.
The left-hand side is

$$L^{(1)} = (V_L, E_L),$$

where

$$V_L = \{x_{ab}, x_c\}, \qquad x_{ab} := \{a, b\}, \qquad x_c := \{c\},$$

and

$$E_L = \{\ell\}, \qquad \ell = \{x_{ab}, x_c\}.$$

The interface is

$$K^{(1)} = (V_K, E_K),$$

where

$$V_K = \{y_{ab}\}, \qquad y_{ab} := \{a, b\}, \qquad E_K = \varnothing.$$

The right-hand side is

$$R^{(1)} = (V_R, E_R),$$

where

$$V_R = \{z_{ab}, z_{cd}\}, \qquad z_{ab} := \{a, b\}, \qquad z_{cd} := \{c, d\},$$

and

$$E_R = \{r_1\}, \qquad r_1 = \{z_{ab}, z_{cd}\}.$$

Define the monomorphisms

$$l : K^{(1)} \hookrightarrow L^{(1)}, \qquad l(y_{ab}) = x_{ab},$$

and

$$r : K^{(1)} \hookrightarrow R^{(1)}, \qquad r(y_{ab}) = z_{ab}.$$

Thus the interface preserves the common supervertex $\{a, b\}$, while the left-hand side interaction

$$\{\,\{a, b\}, \{c\}\,\}$$

is replaced by the right-hand side interaction

$$\{\,\{a, b\}, \{c, d\}\,\}.$$

Step 3: The set of rewrite rules. Let

$$\mathcal{R}^{(1)} = \{\rho\}.$$

Then

$$\mathfrak{H}^{(1)} = \left(\mathcal{H}^{(1)}, \mathcal{R}^{(1)}\right)$$

is a Rewrite 1-SuperHyperGraph.

The supervertex $\{a, b\}$ may be viewed as a stable core team. The rewrite rule ρ replaces a collaboration between the core team $\{a, b\}$ and the singleton team $\{c\}$ by a collaboration between the same core team and the larger grouped team $\{c, d\}$, while preserving the common interface represented by $\{a, b\}$.

Theorem 5.20.9 (Well-definedness of Rewrite n-SuperHyperGraphs)**.** *Let V_0 be a finite nonempty base set, let $n \in \mathbb{N}_0$, and let*

$$\mathcal{H}^{(n)} = (V, E)$$

be a finite n-SuperHyperGraph. Let

$$\mathcal{R}^{(n)}$$

be a finite set such that every element

$$\rho \in \mathcal{R}^{(n)}$$

is a rewrite rule for n-SuperHyperGraphs, i.e. a span

$$\rho = \left(L^{(n)} \xleftarrow{l} K^{(n)} \xrightarrow{r} R^{(n)}\right),$$

where $L^{(n)}, K^{(n)}, R^{(n)}$ are finite n-SuperHyperGraphs and

$$l : K^{(n)} \hookrightarrow L^{(n)}, \qquad r : K^{(n)} \hookrightarrow R^{(n)}$$

are monomorphisms of n-SuperHyperGraphs.

Then the pair

$$\mathfrak{H}^{(n)} = \left(\mathcal{H}^{(n)}, \mathcal{R}^{(n)}\right)$$

is a well-defined Rewrite n-SuperHyperGraph.

Proof. Since $\mathcal{H}^{(n)}$ is assumed to be a finite n-SuperHyperGraph, its underlying data are well-defined: there exists a finite nonempty base set V_0 such that

$$V \subseteq \mathcal{P}^n(V_0)$$

is a finite set of n-supervertices and

$$E \subseteq \mathcal{P}(V) \setminus \{\varnothing\}$$

is a finite set of nonempty n-superhyperedges.

Next, let $\rho \in \mathcal{R}^{(n)}$. By assumption, ρ is a rewrite rule for n-SuperHyperGraphs, hence

$$\rho = \left(L^{(n)} \xleftarrow{l} K^{(n)} \xrightarrow{r} R^{(n)}\right),$$

where $L^{(n)}, K^{(n)}, R^{(n)}$ are finite n-SuperHyperGraphs and the maps

$$l : K^{(n)} \hookrightarrow L^{(n)}, \qquad r : K^{(n)} \hookrightarrow R^{(n)}$$

are monomorphisms of n-SuperHyperGraphs. Therefore each ρ is a well-defined span in the category of finite n-SuperHyperGraphs.

Since $\mathcal{R}^{(n)}$ is assumed finite, it is a well-defined finite set of such rewrite rules. Hence the ordered pair

$$\left(\mathcal{H}^{(n)}, \mathcal{R}^{(n)}\right)$$

consists of:

1. a well-defined finite n-SuperHyperGraph $\mathcal{H}^{(n)}$, and
2. a well-defined finite set $\mathcal{R}^{(n)}$ of rewrite rules for n-SuperHyperGraphs.

This is exactly the data required in the definition of a Rewrite n-SuperHyperGraph. Consequently,

$$\mathfrak{H}^{(n)} = \left(\mathcal{H}^{(n)}, \mathcal{R}^{(n)}\right)$$

is a well-defined Rewrite n-SuperHyperGraph. □

Corollary 5.20.10 (Special case $n = 0$)**.** *When $n = 0$, a Rewrite 0-SuperHyperGraph reduces to a Rewrite HyperGraph.*

Proof. If $n = 0$, then

$$\mathcal{P}^0(V_0) = V_0,$$

so an 0-SuperHyperGraph is just an ordinary HyperGraph. Likewise, a rewrite rule for 0-SuperHyperGraphs is precisely a rewrite rule for HyperGraphs. Therefore the pair

$$\mathfrak{H}^{(0)} = \left(\mathcal{H}^{(0)}, \mathcal{R}^{(0)}\right)$$

is exactly a Rewrite HyperGraph. □

5.21 Uncertain SuperHyperGraph

An Uncertain Set assigns to each element a degree from an uncertainty model, unifying fuzzy, intuitionistic, neutrosophic and plithogenic frameworks [267, 268]. An Uncertain Graph is a graph where vertices or edges carry degrees in an uncertainty model, subsuming fuzzy, intuitionistic, neutrosophic. An Uncertain HyperGraph assigns uncertainty-model degrees to vertices and hyperedges in a hypergraph, modeling complex higher-order connections under incomplete information. An Uncertain SuperHyperGraph equips each supervertex and superedge in an n-SuperHyperGraph with uncertainty-model degrees, handling hierarchical uncertainty systematically and rigorously [66].

Definition 5.21.1 (Uncertain Model)**.** [267] Let U denote the class of all *uncertain models*. Each $M \in U$ is specified by

- a nonempty set $\mathrm{Dom}(M) \subseteq [0, 1]^k$ of *admissible degree tuples* for some fixed integer $k \geq 1$;
- model–specific algebraic or geometric constraints on elements of $\mathrm{Dom}(M)$ (for example, $\mu + \nu \leq 1$ in the intuitionistic fuzzy case, or $T + I + F \leq 3$ in the neutrosophic case).

Typical examples include:

- Fuzzy model: $\mathrm{Dom}(M) = [0, 1]$;
- Intuitionistic fuzzy model: $\mathrm{Dom}(M) = \{(\mu, \nu) \in [0, 1]^2 \mid \mu + \nu \leq 1\}$;
- Neutrosophic model: $\mathrm{Dom}(M) = \{(T, I, F) \in [0, 1]^3 \mid 0 \leq T + I + F \leq 3\}$;
- Plithogenic model, and many other extensions.

Definition 5.21.2 (Uncertain Set (U-Set))**.** [267] Let X be a nonempty universe, and let M be a fixed uncertain model with degree–domain $\mathrm{Dom}(M) \subseteq [0, 1]^k$. An *Uncertain Set of type M* (or *U-Set* for short) on X is a pair

$$\mathcal{U} = (X, \mu_M),$$

where

$$\mu_M : X \longrightarrow \mathrm{Dom}(M)$$

is called the *uncertainty–degree function* (or membership map) of $\mathcal{U}$.

For $x \in X$, the value $\mu_M(x) \in \mathrm{Dom}(M)$ encodes the degree(s) to which x belongs to the uncertain set, according to the model M.

Remark 5.21.3. *Special cases:*

- *If M is the fuzzy model and* $\mathrm{Dom}(M) = [0, 1]$*, then $\mu_M : X \to [0, 1]$ is a usual fuzzy membership function and $\mathcal{U}$ is a fuzzy set.*
- *If M is neutrosophic, then $\mu_M(x) = (T(x), I(x), F(x))$ gives a neutrosophic set.*

- *Other choices of M recover intuitionistic fuzzy sets, picture fuzzy sets, plithogenic sets, and so on.*

Definition 5.21.4 (Uncertain n-SuperHyperGraph)**.** [66] Let V_0 be a finite base set and let $n \in \mathbb{N}_0$. Assume that an n-SuperHyperGraph on V_0 is given by

$$\mathrm{SHG}^{(n)} = (V_n, E),$$

where

$$\emptyset \neq V_n \subseteq \mathcal{P}^n(V_0) \quad \text{and} \quad \emptyset \neq E \subseteq \mathcal{P}(V_n) \setminus \{\emptyset\},$$

so that each n-superedge $e \in E$ is a nonempty subset of the n-supervertex set V_n.

Let M be a fixed uncertain model with degree–domain $\mathrm{Dom}(M) \subseteq [0,1]^k$. An *Uncertain n-SuperHyperGraph of type M* is a triple

$$\mathcal{S}_M^{(n)} = (V_n, E, \mu_M),$$

where

$$\mu_M : V_n \cup E \longrightarrow \mathrm{Dom}(M)$$

assigns to each n-supervertex $v \in V_n$ and each n-superedge $e \in E$ an uncertainty degree $\mu_M(v)$ or $\mu_M(e)$ in $\mathrm{Dom}(M)$.

Any additional relations between the degrees of n-superedges and the degrees of the n-supervertices they contain (for example, model- specific bounds or aggregations) are imposed by the chosen uncertain model M and are not fixed at the level of this general definition.

For $n = 0$ and $V_0 = V_n$, the above notion reduces to an Uncertain HyperGraph of type M.

Remark 5.21.5. *Particular choices of the model M recover well–known uncertain SuperHyperGraph types:*

- *Fuzzy n-SuperHyperGraphs (when M is fuzzy);*
- *Intuitionistic fuzzy, neutrosophic, and plithogenic n-SuperHyperGraphs for the corresponding models M;*
- *More exotic variants (e.g. q-rung orthopair, picture fuzzy, refined neutrosophic) are obtained by choosing the appropriate degree–domain* $\mathrm{Dom}(M)$.

Regarding the catalogue of uncertainty-superhypergraph families (Uncertain n-SuperHyperGraphs) by the dimension k of the degree-domain $\mathrm{Dom}(M) \subseteq [0,1]^k$*, we list them in Table 5.2.*

Table 5.2: A catalogue of uncertainty-superhypergraph families (Uncertain n-SuperHyperGraphs) by the dimension k of the degree-domain $\mathrm{Dom}(M) \subseteq [0,1]^k$.

k	Representative uncertainty-superhypergraph family (type M with $\mathrm{Dom}(M) \subseteq [0,1]^k$)
1	*Fuzzy n-SuperHyperGraph[41, 269]:* $\mu_M : V_n \cup E \to [0,1]$.
2	*Intuitionistic-fuzzy n-SuperHyperGraph[270, 269]:* $\mu_M : V_n \cup E \to [0,1]^2$ (e.g., (membership, non-membership)).
3	*Neutrosophic n-SuperHyperGraph [51, 271, 52]:* $\mu_M : V_n \cup E \to [0,1]^3$ (e.g., (T, I, F)).
4	*Quadripartitioned / four-component uncertainty n-SuperHyperGraph:* $\mu_M : V_n \cup E \to [0,1]^4$.
k	*k-component uncertainty n-SuperHyperGraph (cf. Plithogenic n-SuperHyperGraph [55, 54, 272]):* $\mu_M : V_n \cup E \to \mathrm{Dom}(M) \subseteq [0,1]^k$ (model-specific semantics).

5.22 Functorial SuperHyperGraph

A Functorial Set is a functor assigning each object a set and pushing elements along structure-preserving morphisms in a category [267]. A Functorial Graph functorially assigns each object a graph and maps graph homomorphisms along morphisms, preserving composition and identities everywhere. A Functorial HyperGraph assigns each object a hypergraph and transports hyperedges via hypergraph homomorphisms induced by morphisms, respecting categorical composition. A Functorial SuperHyperGraph associates each object with a superhypergraph and sends morphisms to homomorphisms preserving supervertices, superedges, and hierarchical structure.

Definition 5.22.1 (Functorial Set)**.** [267] Let $\mathcal{C}$ be a category and let

$$F : \mathcal{C} \longrightarrow \mathbf{Set}$$

be a covariant functor.

We call F a *Functorial Set* on $\mathcal{C}$. For each object $X \in \mathrm{Ob}(\mathcal{C})$, the set

$$F(X)$$

is interpreted as the collection of "F–sets over X", and every element $s \in F(X)$ is an individual F–set based at X.

Every morphism $f : X \to Y$ in $\mathcal{C}$ induces a *pushforward*

$$F(f) : F(X) \longrightarrow F(Y), \qquad s \longmapsto F(f)(s),$$

and the usual functoriality conditions

$$F(\mathrm{id}_X) = \mathrm{id}_{F(X)}, \qquad F(g \circ f) = F(g) \circ F(f)$$

hold for all composable morphisms $X \xrightarrow{f} Y \xrightarrow{g} Z$.

Definition 5.22.2 (Functorial SuperHyperGraph)**.** Fix an integer $n \geq 1$. Let $\mathbf{SHGraph}_n$ denote the category whose objects are finite level-n SuperHyperGraphs. Concretely, an object is a triple

$$\mathsf{SH} = (V_0, V, E),$$

where

- V_0 is a finite base set;
- $V \subseteq \mathcal{P}^n(V_0)$ is a nonempty set of n-supervertices;
- $E \subseteq \mathcal{P}(V) \setminus \{\emptyset\}$ is a nonempty family of n-superedges, each superedge being a nonempty subset of V.

Thus the supervertices live at the n-th iterated powerset level, while the superedges are ordinary (nonempty) subsets of the supervertex set V.

A morphism

$$\Phi : (V_0, V, E) \longrightarrow (V_0', V', E')$$

in $\mathbf{SHGraph}_n$ is a *superhypergraph homomorphism*, i.e. a base map $\phi_0 : V_0 \to V_0'$ such that the induced map on the n-th iterated powerset

$$\phi_n := \mathcal{P}^n(\phi_0) : \mathcal{P}^n(V_0) \longrightarrow \mathcal{P}^n(V_0')$$

satisfies

$$\phi_n(V) \subseteq V' \quad \text{and} \quad \phi_n[e] := \{\phi_n(v) \mid v \in e\} \in E' \quad \text{for all } e \in E.$$

Let $\mathcal{C}$ be a category. A *Functorial SuperHyperGraph of level* n on $\mathcal{C}$ is a covariant functor

$$\mathsf{SH} : \mathcal{C} \longrightarrow \mathbf{SHGraph}_n.$$

For each object $X \in \mathrm{Ob}(\mathcal{C})$, the value

$$\mathsf{SH}(X) = (V_0^X, V_X, E_X)$$

is a level-n SuperHyperGraph, and for each morphism $f : X \to Y$ in $\mathcal{C}$, the arrow

$$\mathsf{SH}(f) : \mathsf{SH}(X) \longrightarrow \mathsf{SH}(Y)$$

is a superhypergraph homomorphism in the above sense, satisfying

$$\mathsf{SH}(\mathrm{id}_X) = \mathrm{id}_{\mathsf{SH}(X)}, \qquad \mathsf{SH}(g \circ f) = \mathsf{SH}(g) \circ \mathsf{SH}(f)$$

for all composable f, g in $\mathcal{C}$.

In particular, when $n = 0$ and $V = V_0$, a Functorial SuperHyperGraph reduces to a Functorial HyperGraph.

5.23 Topological SuperHyperGraph

Topological superhypergraph models hierarchical higher-order relations by combining superhypergraph nesting with topological structure, enabling continuous, spatial, and inclusion-based representations faithfully[273].

Definition 5.23.1 (Topological superhypergraph)**.** Let $P \subset \mathbb{R}^2$ be a finite nonempty set, called the base point set, and let

$$\mathcal{P}^0(P) := P, \qquad \mathcal{P}^{n+1}(P) := \mathcal{P}(\mathcal{P}^n(P)) \quad (n \geq 0).$$

A *topological n-superhypergraph* is a pair

$$\mathcal{S} = (V, E)$$

such that

$$V \subseteq \mathcal{P}^n(P), \qquad E \subseteq \mathcal{P}^n(P),$$

together with a topological realization assigning to each element of $V \cup E$ a topological object in $\mathbb{R}^2$ (for example, a point set, closed curve, or closed region), so that the incidence relation is represented by set inclusion of the corresponding realizations.

In particular, when $n = 1$, this reduces to a topological hypergraph.

Example 5.23.2 (A concrete topological 2-superhypergraph)**.** Let

$$P = \{p_1, p_2, p_3\} \subset \mathbb{R}^2,$$

where

$$p_1 = (0, 0), \qquad p_2 = (2, 0), \qquad p_3 = (1, 2).$$

Then

$$\mathcal{P}^1(P) = \mathcal{P}(P), \qquad \mathcal{P}^2(P) = \mathcal{P}(\mathcal{P}(P)).$$

Define

$$v_1 := \{\{p_1\}, \{p_1, p_2\}\}, \qquad v_2 := \{\{p_2\}, \{p_2, p_3\}\},$$

and

$$e := \{\{p_1\}, \{p_1, p_2\}, \{p_2\}, \{p_2, p_3\}\}.$$

Clearly,

$$v_1, v_2, e \in \mathcal{P}^2(P).$$

Now set

$$V := \{v_1, v_2\}, \qquad E := \{e\}.$$

Thus

$$V \subseteq \mathcal{P}^2(P), \qquad E \subseteq \mathcal{P}^2(P).$$

Next, define a realization map

$$\rho : V \cup E \longrightarrow \{\text{topological subspaces of } \mathbb{R}^2\}$$

by

$$\rho(X) := \bigcup_{A \in X} \operatorname{conv}(A),$$

where $\operatorname{conv}(A)$ denotes the convex hull of $A \subseteq P$.

Then

$$\rho(v_1) = \operatorname{conv}(\{p_1\}) \cup \operatorname{conv}(\{p_1, p_2\}),$$

so $\rho(v_1)$ consists of the point p_1 together with the closed line segment joining p_1 and p_2. Similarly,

$$\rho(v_2) = \operatorname{conv}(\{p_2\}) \cup \operatorname{conv}(\{p_2, p_3\}),$$

which consists of the point p_2 together with the closed line segment joining p_2 and p_3. Moreover,

$$\rho(e) = \operatorname{conv}(\{p_1\}) \cup \operatorname{conv}(\{p_1, p_2\}) \cup \operatorname{conv}(\{p_2\}) \cup \operatorname{conv}(\{p_2, p_3\}).$$

Since

$$v_1 \subseteq e \qquad \text{and} \qquad v_2 \subseteq e,$$

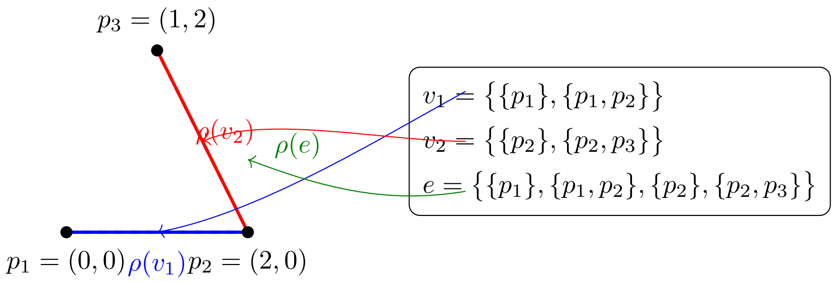


Figure 5.10: A schematic illustration of the concrete topological 2-superhypergraph. The realization $\rho(v_1)$ consists of the point p_1 together with the segment joining p_1 and p_2, the realization $\rho(v_2)$ consists of the point p_2 together with the segment joining p_2 and p_3, and the realization $\rho(e)$ is their union. Thus $\rho(v_1) \subseteq \rho(e)$ and $\rho(v_2) \subseteq \rho(e)$.

it follows that

$$\rho(v_1) \subseteq \rho(e) \qquad \text{and} \qquad \rho(v_2) \subseteq \rho(e).$$

Hence the incidence relation is represented by inclusion of the corresponding topological realizations.

Therefore,

$$\mathcal{S} = (V, E)$$

is a concrete example of a topological 2-superhypergraph.

A schematic illustration of this concrete topological 2-superhypergraph is shown in Fig. 5.10.

5.24 Motif Hypergraphs and Motif SuperHypergraphs

A motif hypergraph represents each motif instance as a hyperedge on its supporting vertices, capturing recurring higher-order patterns and aggregating structurally similar local interactions faithfully. A motif superhypergraph treats motif supports as supervertices and links families of them through higher-order relations, encoding interactions among motif-defined subsets themselves in complex networks.

Definition 5.24.1 (Motif instance)**.** Let

$$M = (V_M, E_M)$$

be a finite graph (respectively, digraph), and let

$$G = (V, E)$$

be a finite graph (respectively, digraph) of the same type.

An *M-instance* in G is an injective map

$$\phi : V_M \to V$$

such that for all $u, v \in V_M$,

$$\{u, v\} \in E_M \implies \{\phi(u), \phi(v)\} \in E$$

in the undirected case, and

$$(u, v) \in E_M \implies (\phi(u), \phi(v)) \in E$$

in the directed case.

We write

$$\mathrm{Inst}_M(G)$$

for the set of all M-instances in G, and for $\phi \in \mathrm{Inst}_M(G)$ we define its *vertex support* by

$$\mathrm{supp}(\phi) := \phi(V_M) \subseteq V.$$

If one wishes to work with induced motifs, the above implication is replaced by

$$\{u, v\} \in E_M \iff \{\phi(u), \phi(v)\} \in E$$

(or the directed analogue).

Definition 5.24.2 (Motif Hypergraph)**.** Let M and G be as above, and let

$$\omega : \mathrm{Inst}_M(G) \to \mathbb{R}_{\geq 0}$$

be a nonnegative weight function on motif instances.

The *motif hypergraph* of G with respect to M is the weighted hypergraph

$$\mathcal{H}_M(G) = \big(V, \mathcal{E}_M, w_M\big),$$

where

$$\mathcal{E}_M := \big\{\mathrm{supp}(\phi) \ : \ \phi \in \mathrm{Inst}_M(G)\big\} \subseteq \mathcal{P}^*(V),$$

and for each $e \in \mathcal{E}_M$,

$$w_M(e) := \sum_{\substack{\phi \in \mathrm{Inst}_M(G) \\ \mathrm{supp}(\phi) = e}} \omega(\phi).$$

Thus each hyperedge is the vertex set of an M-instance, and if several motif instances have the same support, their weights are aggregated.

In the unweighted case, one usually takes

$$\omega(\phi) = 1 \qquad (\phi \in \mathrm{Inst}_M(G)),$$

so that $w_M(e)$ counts the number of M-instances having support e.

Example 5.24.3 (A concrete Motif Hypergraph)**.** Let

$$M = K_3$$

be the triangle graph with vertex set

$$V_M = \{x_1, x_2, x_3\}$$

and edge set

$$E_M = \big\{\{x_1, x_2\}, \{x_2, x_3\}, \{x_1, x_3\}\big\}.$$

Let

$$G = (V, E)$$

be the graph with

$$V = \{1, 2, 3, 4\}$$

and

$$E = \big\{\{1, 2\}, \{2, 3\}, \{1, 3\}, \{2, 4\}, \{3, 4\}\big\}.$$

Thus G contains exactly two triangle subgraphs: one on $\{1, 2, 3\}$ and one on $\{2, 3, 4\}$.

Define two M-instances in G by

$$\phi_1(x_1) = 1, \qquad \phi_1(x_2) = 2, \qquad \phi_1(x_3) = 3,$$

and

$$\phi_2(x_1) = 2, \qquad \phi_2(x_2) = 3, \qquad \phi_2(x_3) = 4.$$

Then

$$\mathrm{supp}(\phi_1) = \{1, 2, 3\}, \qquad \mathrm{supp}(\phi_2) = \{2, 3, 4\}.$$

Take the unweighted case

$$\omega(\phi) = 1 \qquad (\phi \in \mathrm{Inst}_M(G)).$$

Hence the motif hyperedge set is

$$\mathcal{E}_M = \big\{\{1, 2, 3\}, \{2, 3, 4\}\big\},$$

and the weight function is given by

$$w_M(\{1, 2, 3\}) = 1, \qquad w_M(\{2, 3, 4\}) = 1.$$

Therefore, the motif hypergraph of G with respect to the triangle motif M is

$$\mathcal{H}_M(G) = \big(\{1, 2, 3, 4\}, \{\{1, 2, 3\}, \{2, 3, 4\}\}, w_M\big).$$

This hypergraph records the two triangle motifs of G as hyperedges on their supporting vertex sets. A schematic illustration of the motif hypergraph for the triangle motif $M = K_3$ is shown in Fig. 5.11.

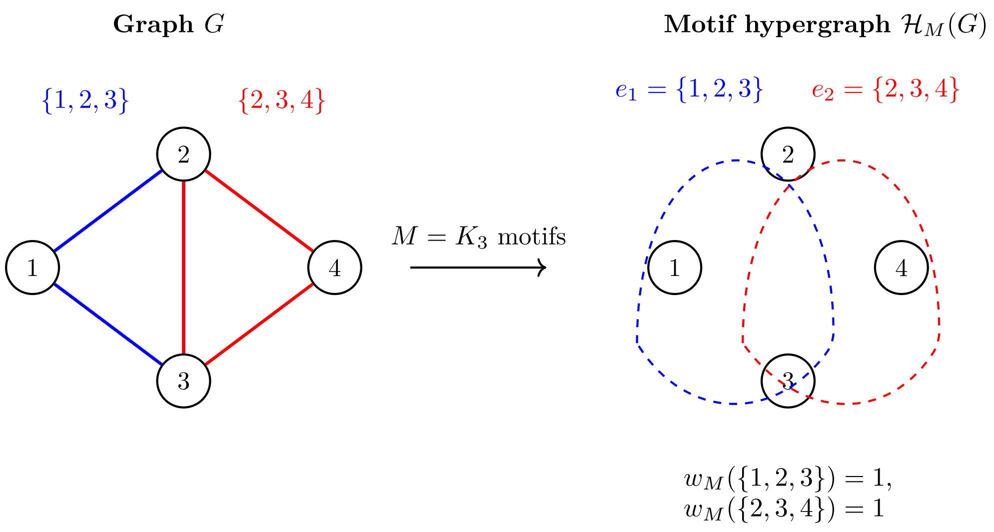


Figure 5.11: A schematic illustration of the motif hypergraph associated with the triangle motif $M = K_3$. The graph G contains exactly two triangle instances, on the vertex sets $\{1,2,3\}$ and $\{2,3,4\}$, and these are recorded as hyperedges in the motif hypergraph $\mathcal{H}_M(G)$.

Definition 5.24.4 (Motif SuperHyperGraph)**.** Let M and $G = (V, E)$ be as above. Define the set of *motif supervertices* by

$$\mathcal{V}_M := \{\mathrm{supp}(\phi) \ : \ \phi \in \mathrm{Inst}_M(G)\} \subseteq \mathcal{P}^*(V).$$

Hence every element of $\mathcal{V}_M$ is itself a subset of the base set V.

Let

$$R : \mathcal{P}^*(\mathcal{V}_M) \to \{\mathrm{true}, \mathrm{false}\}$$

be a prescribed higher-order incidence rule on nonempty families of motif supports. Define

$$\mathcal{F}_M := \{\mathcal{A} \in \mathcal{P}^*(\mathcal{V}_M) \ : \ R(\mathcal{A}) \text{ holds}\}.$$

Then the pair

$$\mathfrak{S}_M(G; R) := (\mathcal{V}_M, \mathcal{F}_M)$$

is called the *motif SuperHyperGraph* of G with respect to M and R.

Equivalently, $\mathfrak{S}_M(G; R)$ is a 1-SuperHyperGraph over the base set V, since

$$\mathcal{V}_M \subseteq \mathcal{P}(V) \qquad \text{and} \qquad \mathcal{F}_M \subseteq \mathcal{P}^*(\mathcal{V}_M).$$

A canonical choice of the rule R is the *overlap rule*

$$R_{\mathrm{ov}}(\mathcal{A}) \iff \bigcap_{S \in \mathcal{A}} S \neq \emptyset.$$

In that case,

$$\mathcal{F}_M^{\mathrm{ov}} = \{\mathcal{A} \in \mathcal{P}^*(\mathcal{V}_M) \ : \ \bigcap_{S \in \mathcal{A}} S \neq \emptyset\},$$

and the resulting motif SuperHyperGraph connects families of motif instances having a common vertex.

Remark 5.24.5. *The motif hypergraph records* which vertex subsets support motif instances*, whereas the motif SuperHyperGraph records* higher-order relations among those motif-supported subsets themselves*. Hence the former is a hypergraph on V, while the latter is a SuperHyperGraph whose vertices are motif supports.*

Example 5.24.6 (A concrete Motif SuperHyperGraph)**.** Let $M = K_3$ and let $G = (V, E)$ be the same graph as in the previous example, namely

$$V = \{1, 2, 3, 4\},$$

and

$$E = \{\{1,2\}, \{2,3\}, \{1,3\}, \{2,4\}, \{3,4\}\}.$$

As above, G has two triangle motif supports:

$$S_1 := \{1, 2, 3\}, \qquad S_2 := \{2, 3, 4\}.$$

Hence the set of motif supervertices is

$$\mathcal{V}_M = \{S_1, S_2\}.$$

Now choose the canonical overlap rule

$$R_{\mathrm{ov}}(\mathcal{A}) \iff \bigcap_{S \in \mathcal{A}} S \neq \emptyset.$$

Since

$$S_1 \cap S_2 = \{2, 3\} \neq \emptyset,$$

the family

$$\{S_1, S_2\}$$

satisfies the overlap rule. Also, each singleton family satisfies the rule because

$$S_1 \neq \emptyset, \qquad S_2 \neq \emptyset.$$

Therefore,

$$\mathcal{F}_M^{\mathrm{ov}} = \big\{\{S_1\}, \{S_2\}, \{S_1, S_2\}\big\}.$$

Thus the motif SuperHyperGraph is

$$\mathfrak{S}_M(G; R_{\mathrm{ov}}) = \big(\{S_1, S_2\}, \{\{S_1\}, \{S_2\}, \{S_1, S_2\}\}\big).$$

Equivalently,

$$\mathfrak{S}_M(G; R_{\mathrm{ov}}) = \big(\{\{1,2,3\}, \{2,3,4\}\}, \{\{\{1,2,3\}\}, \{\{2,3,4\}\}, \{\{1,2,3\}, \{2,3,4\}\}\}\big).$$

In particular, the nontrivial superhyperedge

$$\{S_1, S_2\}$$

expresses that the two triangle motifs overlap on the common vertex set $\{2, 3\}$.

5.25 Molecular SuperHyperGraphs

A molecular graph represents a molecule as a labeled graph in which vertices correspond to atoms and edges correspond to covalent bonds [274, 130, 275, 107, 276]. A molecular hypergraph generalizes this representation by allowing hyperedges to connect multiple atoms simultaneously, thereby capturing functional groups, aromatic rings, delocalized electron systems, and other multi-atom chemical interactions [277, 278, 279, 280]. A molecular SuperHyperGraph extends this idea further by organizing atoms, bonds, fragments, and larger molecular units across iterated powerset levels, thus providing a hierarchical framework for representing complex chemical structures, multiscale motifs, and overlapping functional contexts [281, 282]. Related notions include chemical graphs [283, 284, 285], chemical hypergraphs [286, 287, 288], chemical SuperHyperGraphs [289, 290], goal-directed molecular graphs [291, 292], and chemical reaction networks [293, 294, 295], all of which arise in computational chemistry, cheminformatics, and reaction-mechanism modeling.

Definition 5.25.1 (Molecular Graph)**.** (cf. [274]) A *molecular graph* is a labeled simple graph

$$G = (V, E, \ell_V, \ell_E),$$

where

- V is a finite set of *atoms*;
- $E \subseteq \{\{u, v\} \mid u, v \in V,\ u \neq v\}$ is the finite set of *covalent bonds*;
- $\ell_V : V \to \mathcal{L}_V$ assigns to each vertex $v \in V$ its *atomic label* (for example, an element symbol such as C, H, or O);
- $\ell_E : E \to \mathcal{L}_E$ assigns to each edge $e \in E$ its *bond label* (for example, single, double, or triple).

Thus, vertices represent atoms, edges represent bonds, and the labeling maps encode atomic and bond types.

Definition 5.25.2 (Molecular n-SuperHyperGraph)**.** [281, 282] Let V_0 be a finite set of *bond identifiers* associated with a molecule. Define the iterated powersets by

$$\mathrm{P}_0(V_0) := V_0, \qquad \mathrm{P}_{k+1}(V_0) := \mathrm{P}\big(\mathrm{P}_k(V_0)\big) \quad (k \geq 0),$$

where $\mathrm{P}(\cdot)$ denotes the ordinary powerset operator.

Fix an integer $n \geq 0$. A *molecular n-SuperHyperGraph* on the base set V_0 is a quintuple

$$H^{(n)} = \big(V_H, E_H, \partial, \ell_V^H, \ell_E^H\big),$$

where

- $V_H \subseteq \mathrm{P}_n(V_0)$ is a finite set of *n-supervertices*, each representing a possibly nested collection of bond identifiers up to level n;
- E_H is a finite set of *n-superedges*;
- $\partial : E_H \to \mathcal{P}^*(V_H)$ is the incidence map, where

$$\mathcal{P}^*(V_H) := \mathcal{P}(V_H) \setminus \{\emptyset\};$$

  for each $e \in E_H$, the set $\partial(e) \subseteq V_H$ is the family of n-supervertices incident with the n-superedge e;
- $\ell_V^H : V_H \to \mathcal{L}_V$ assigns to each n-supervertex $v \in V_H$ a *vertex label* (for example, a bond-pattern type, functional-group name, or moiety type);
- $\ell_E^H : E_H \to \mathcal{L}_E$ assigns to each n-superedge $e \in E_H$ an *edge label* (for example, an atom symbol, fragment name, or whole-molecule / functional-unit identifier).

The underlying n-SuperHyperGraph of $H^{(n)}$ is the triple

$$(V_H, E_H, \partial).$$

When $n = 0$, the condition $V_H \subseteq \mathrm{P}_0(V_0) = V_0$ implies that each vertex corresponds to a single bond identifier, and the structure reduces to a labeled molecular hypergraph on V_0.

Example 5.25.3 (A concrete molecular 1-SuperHyperGraph for ethanol)**.** Consider the molecule ethanol, with bond identifiers

$$V_0 = \{b_1, b_2, b_3\},$$

where

$$b_1 = \text{the bond } \mathrm{C}_1{-}\mathrm{C}_2, \qquad b_2 = \text{the bond } \mathrm{C}_2{-}\mathrm{O}, \qquad b_3 = \text{the bond } \mathrm{O}{-}\mathrm{H}.$$

Then

$$\mathrm{P}_0(V_0) = V_0, \qquad \mathrm{P}_1(V_0) = \mathrm{P}(V_0).$$

Define the following 1-supervertices:

$$v_1 := \{b_1, b_2\}, \qquad v_2 := \{b_2, b_3\}, \qquad v_3 := \{b_1, b_2, b_3\}.$$

Thus,

$$V_H := \{v_1, v_2, v_3\} \subseteq \mathrm{P}_1(V_0).$$

Let

$$E_H := \{e_1, e_2\},$$

and define the incidence map

$$\partial : E_H \to \mathcal{P}^*(V_H)$$

by

$$\partial(e_1) = \{v_1, v_2\}, \qquad \partial(e_2) = \{v_1, v_2, v_3\}.$$

Next, define the vertex-label map $\ell_V^H : V_H \to \mathcal{L}_V$ by

$$\ell_V^H(v_1) = \text{“C–C–O backbone fragment”},$$

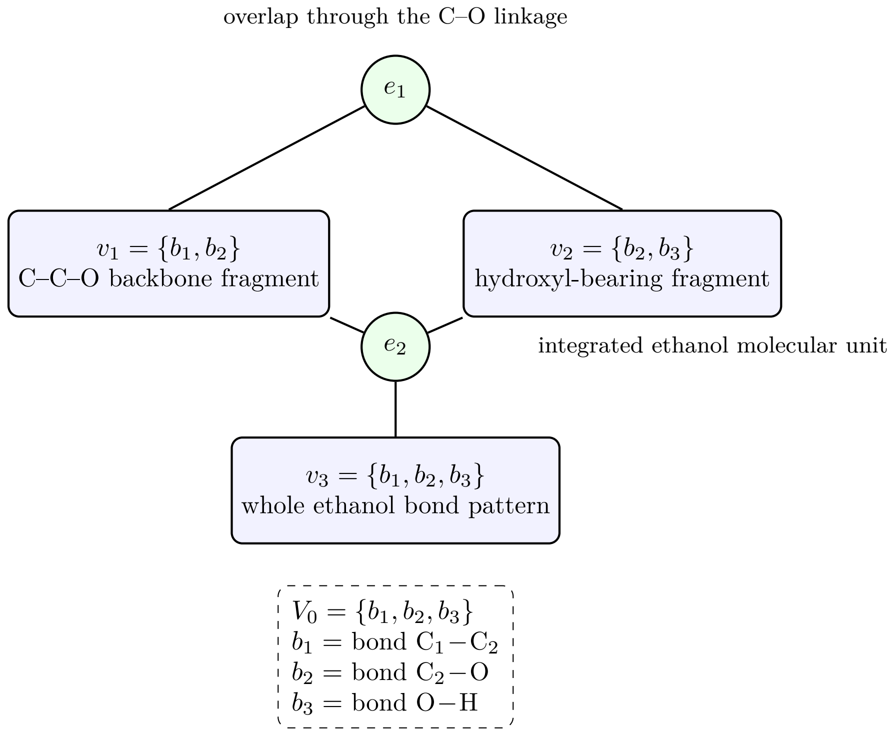


Figure 5.12: A TikZ illustration of the concrete molecular 1-SuperHyperGraph for ethanol. The three 1-supervertices represent overlapping bond-based fragments of ethanol, while the two superedges encode higher-order incidence relations among them.

$$\ell_V^H(v_2) = \text{“hydroxyl-bearing fragment”},$$

$$\ell_V^H(v_3) = \text{“whole ethanol bond pattern”},$$

and define the edge-label map $\ell_E^H : E_H \to \mathcal{L}_E$ by

$$\ell_E^H(e_1) = \text{“overlap through the C–O linkage”}, \qquad \ell_E^H(e_2) = \text{“integrated ethanol molecular unit”}.$$

Hence

$$H^{(1)} = \big(V_H, E_H, \partial, \ell_V^H, \ell_E^H\big)$$

is a molecular 1-SuperHyperGraph on the base set V_0.

In this example, v_1 represents the carbon-chain fragment adjacent to oxygen, v_2 represents the hydroxyl-side fragment, and v_3 represents the full bond configuration of ethanol. The superedge e_1 records the higher-order relation between the two overlapping local fragments, while e_2 collects them together with the whole-molecule bond pattern into a single molecular unit. A schematic illustration of this molecular 1-SuperHyperGraph for ethanol is shown in Fig. 5.12.

6 Discussions: Complete Higher-Graphic Structure

In this chapter, we aim to examine whether a unifying framework can be developed by integrating the concepts introduced so far.

6.1 Complete Higher-Graphic Structure

A Complete Higher-Graphic Structure (CHGS) is a unified typed framework for modeling higher-order networks by combining multiway relations, compositional operations, attributes, weights, and closure mechanisms across contexts, levels, and semantics.

Definition 6.1.1 (Complete Higher-Graphic Structure (CHGS))**.** Let I be a (possibly finite) set of *contexts* and let S be a set of *sorts*. A *Complete Higher-Graphic Structure* (abbreviated *CHGS*) is a modular typed structure

$$\mathfrak{H} = \big(I,\ S,\ X,\ \Sigma_{\mathrm{rel}},\ \Sigma_{\mathrm{op}},\ [\![-]\!],\ \mathcal{E},\ \mathcal{A},\ \mathcal{W},\ \mathcal{C}\big),$$

where the components are as follows.

1. **Context–sort carriers.** For each profile $p = (i, s) \in I \times S$, one specifies a set

$$X_p = X_{i,s}.$$

   We write $X = \{X_{i,s}\}_{(i,s)\in I\times S}$. Contexts may encode, for example, grades, layers, times, scales, or modalities.

2. **Relational signature.** Σ_{rel} is a set of *relation symbols*. Each $\sigma \in \Sigma_{\mathrm{rel}}$ is assigned:
   - a finite input profile list

$$\mathrm{prof}(\sigma) = (p_1, \dots, p_m) \in (I \times S)^m \quad (m \geq 1),$$

   - and optionally a symmetry subgroup $G_\sigma \leq S_m$ (to encode unordered or partially symmetric interactions).

3. **Operational signature.** Σ_{op} is a set of *operation symbols*. Each $\omega \in \Sigma_{\mathrm{op}}$ is assigned:
   - a finite input profile list

$$\mathrm{in}(\omega) = (p_1, \dots, p_m) \in (I \times S)^m \quad (m \geq 0),$$

   - an output profile

$$\mathrm{out}(\omega) = q \in I \times S,$$

   - and optionally a symmetry subgroup $G_\omega \leq S_m$.

   These operations model compositional mechanisms (e.g. gluing, substitution, wiring, face/degeneracy maps, etc.).

4. **Interpretation of symbols.** The interpretation $[\![-]\!]$ assigns:
   - to each $\sigma \in \Sigma_{\mathrm{rel}}$ with $\mathrm{prof}(\sigma) = (p_1, \dots, p_m)$, a relation

$$[\![\sigma]\!] \subseteq X_{p_1} \times \cdots \times X_{p_m};$$

   - to each $\omega \in \Sigma_{\mathrm{op}}$ with $\mathrm{in}(\omega) = (p_1, \dots, p_m)$ and $\mathrm{out}(\omega) = q$, a (possibly partial) map

$$[\![\omega]\!] : X_{p_1} \times \cdots \times X_{p_m} \rightharpoonup X_q.$$

   If a symmetry subgroup G_σ or G_ω is specified, the interpretation is required to be invariant/equivariant under the corresponding coordinate permutations (in the standard sense).

5. **Equational / coherence layer (optional but allowed).** $\mathcal{E}$ is a set of well-typed equations between terms generated from Σ_{op}. These equations may impose, for example, associativity, unitality, operadic equivariance, monoidal coherence, or simplicial identities.

6. **Attribute layer (optional but allowed).** $\mathcal{A}$ is a family of additional typed maps (possibly partial), each of the form
$$\alpha : X_{p_1} \times \cdots \times X_{p_m} \rightharpoonup D_\alpha,$$
where D_α is a specified codomain set. This layer is used to encode extra structure such as:
   - support/flattening maps (for supervertices or nested objects),
   - labels and types,
   - grading maps,
   - geometric realizations,
   - or coalgebraic neighborhood maps (by taking $D_\alpha = F(X_p)$ for a chosen endofunctor F).

7. **Weight / tensor layer (optional but allowed).** $\mathcal{W}$ consists of a coefficient set (typically a semiring or ring) K, together with weight assignments on chosen relations or tuples, e.g.
$$w_\sigma : [\![\sigma]\!] \to K,$$
or equivalently full tensors on Cartesian powers when convenient (as in adjacency-tensor models).

8. **Closure layer (optional but allowed).** $\mathcal{C}$ is a family of closure operators on selected carrier sets (or unions of carrier sets), e.g.
$$\mathrm{cl} : \mathcal{P}(Y) \to \mathcal{P}(Y),$$
where Y is a chosen set built from the $X_{i,s}$, satisfying extensivity, monotonicity, and idempotence. This layer encodes implication/forcing semantics.

A *network event* in $\mathfrak{H}$ is either (i) a tuple belonging to some interpreted relation $[\![\sigma]\!]$, or (ii) the value of a well-typed operation term built from Σ_{op}, considered modulo the equations $\mathcal{E}$.

A CHGS is called *finite* if all carrier sets $X_{i,s}$ are finite and all listed signatures/modules are finite.

Remark 6.1.2 (Why this is "complete" for higher-order network modeling)**.** *The word* complete *is used here in the sense of an* umbrella formalism*: a CHGS simultaneously allows*

- ***typed multiway relations*** *(hypergraph-, relational-, or tuple-style modeling),*
- ***compositional operations*** *(operadic/monoidal/simplicial/categorical constructions),*
- ***attributes*** *(supports, nested neighborhoods, geometry, labels),*
- ***weights/tensors*** *(quantitative higher-order adjacency),*
- *and* ***closure operators*** *(implication/forcing semantics).*

By switching modules on or off, and by choosing appropriate signatures, one recovers many concrete higher-order graph frameworks as special cases.

Theorem 6.1.3 (Expressive generality of CHGS for the notions treated in this manuscript)**.** *Each of the higher-order frameworks treated in this manuscript can be represented as a specialization of Definition 6.1.1 (up to the usual notion of isomorphism/semantic equivalence appropriate to that framework).*

In particular, this includes:

1. *graph, hypergraph, multigraph, and hypergraph-type variants;*
2. *superhypergraph, graded/hierarchical/recursive/nested superhypergraph-type variants;*
3. *set-family and subset-combinatorial graph constructions (Johnson, Kneser, power-set-based, etc.);*
4. *simplicial, cell, CW, and polyhedral-complex-based structures;*
5. *multilayer and temporal networks;*

6. *factor graphs, Tanner graphs, and related bipartite constraint models;*
7. *relational-arity models (RAG);*
8. *adjacency-tensor models (ATN);*
9. *closure-based implication models (CIG);*
10. *coalgebraic nested-neighborhood models (CNNG);*
11. *operadic interaction models (OIG);*
12. *symmetric monoidal wiring models (SMWG).*

Proof. We give a uniform encoding scheme.

(A) Hypergraph-, graph-, and relation-based models. Choose one context $I = \{*\}$, one or more sorts (e.g. vertices, edge-objects, supervertices), and place the interactions in Σ_{rel}.

- Ordinary directed/undirected graphs arise from binary relation symbols (with or without symmetry).
- Hypergraphs arise either from k-ary relations or from an edge-object sort plus an incidence relation.
- Relational-Arity Graphs (RAG) are exactly multi-sorted (or single-sorted) relational signatures with interpreted relations of fixed arities.

(B) Superhypergraph and nested/hierarchical variants. Use multiple sorts and/or contexts to represent levels (e.g. grade $0, 1, \ldots, n$), and use the attribute layer $\mathcal{A}$ for support/flattening maps into finite subsets of lower-level carriers (or into iterated powersets). Cross-level and within-level superhyperedges are then simply typed relations in Σ_{rel}.

(C) Topological and complex-based frameworks. Use sorts indexed by dimension (vertices, edges, faces, cells, ...), and include face/incidence maps in Σ_{op} or Σ_{rel}. Impose simplicial, cellular, or CW-type axioms via equations $\mathcal{E}$.

(D) Temporal and multilayer networks. Use contexts I to encode time, layers, or their products. Interactions are then typed by profiles (i, s), allowing intracontext and intercontext relations (e.g. interlayer coupling, temporal transition, multiplex constraints).

(E) Tensor-based models (ATN). Keep the relational profile(s) for k-way interactions and activate the weight/tensor layer $\mathcal{W}$, either as weights on tuples or as full tensors on Cartesian powers. Symmetry constraints (undirected case) are imposed by the symmetry metadata of the corresponding relation symbols.

(F) Closure-implication models (CIG). Choose a carrier Y (typically a vertex set or a selected union of carriers) and activate the closure layer $\mathcal{C}$ with a closure operator cl. Forcing relations are then read from membership $a \in \mathrm{cl}(S) \setminus S$.

(G) Coalgebraic nested-neighborhood models (CNNG). Represent vertices in a carrier X_p, and place the coalgebraic neighborhood assignment in the attribute layer $\mathcal{A}$ as a map

$$\nu : X_p \to F(X_p),$$

where F is the chosen endofunctor (e.g. $F = \mathcal{P}_f^r$).

(H) Operadic and monoidal compositional models (OIG/SMWG). Treat operations/morphisms themselves as elements of suitable carriers (sorts indexed by input/output types), and put composition/substitution/tensor/sym maps into Σ_{op}. The corresponding axioms (operad axioms, monoidal coherence, etc.) are imposed in $\mathcal{E}$.

Thus each listed framework appears as a CHGS with a particular choice of contexts, sorts, signatures, interpretations, and optional modules. Hence CHGS is expressive enough to serve as a common host formalism for all of them. □

Example 6.1.4 (A CHGS combining superlevel, relational, tensor, closure, and operadic features)**.** We construct a finite CHGS that simultaneously exhibits:

- a supervertex-like object and a cross-level hyperedge,
- binary and ternary interactions (RAG/ATN flavor),

- a closure implication (CIG flavor),
- a nested-neighborhood map (CNNG flavor),
- and a typed compositional operation (OIG flavor).

(1) Contexts and sorts. Take a single context

$$I = \{*\},$$

and the sort set

$$S = \{\mathsf{X}, \mathsf{H}, \mathsf{Raw}, \mathsf{Clean}, \mathsf{Model}\}.$$

Here:

- X will contain graph-like entities (base and super-level objects),
- H will contain hyperedge-objects,
- $\mathsf{Raw}, \mathsf{Clean}, \mathsf{Model}$ are workflow types.

(2) Carrier sets. Define

$$X_{*,\mathsf{X}} = \{1, 2, 3, \beta\}, \qquad X_{*,\mathsf{H}} = \{\varepsilon_1, \varepsilon_2\},$$

$$X_{*,\mathsf{Raw}} = \{r_1, r_2\}, \qquad X_{*,\mathsf{Clean}} = \{c_1, c_2\}, \qquad X_{*,\mathsf{Model}} = \{m\}.$$

Intuitively, $1, 2, 3$ are base vertices and β is a supervertex.

(3) Relational signature and interpretation. Let Σ_{rel} contain the following symbols:

1. a unary symbol Base on profile $(*, \mathsf{X})$,
2. a unary symbol Super on profile $(*, \mathsf{X})$,
3. a binary incidence symbol Inc on profile $(*, \mathsf{H}), (*, \mathsf{X})$,
4. a binary interaction symbol R_2 on profile $(*, \mathsf{X}), (*, \mathsf{X})$,
5. a ternary interaction symbol R_3 on profile $(*, \mathsf{X}), (*, \mathsf{X}), (*, \mathsf{X})$.

Interpret them by

$$[\![\mathrm{Base}]\!] = \{1, 2, 3\}, \qquad [\![\mathrm{Super}]\!] = \{\beta\},$$

$$[\![\mathrm{Inc}]\!] = \{(\varepsilon_1, \beta), (\varepsilon_1, 3), (\varepsilon_2, 1), (\varepsilon_2, 2)\}.$$

Thus ε_1 is a *cross-level* hyperedge incident to β and 3, while ε_2 is a base-level hyperedge incident to 1 and 2.
Next, define

$$[\![R_2]\!] = \{(1, 2), (2, 3)\}, \qquad [\![R_3]\!] = \{(1, 2, 3)\}.$$

These provide pairwise and ternary interactions on the X-carrier.

(4) Attribute layer (support and nested neighborhoods). Add an attribute map

$$\mathrm{supp}: \ X_{*,\mathsf{X}} \rightharpoonup \mathcal{P}_f(\{1, 2, 3\}),$$

defined by

$$\mathrm{supp}(\beta) = \{1, 2\},$$

and undefined on $1, 2, 3$. This records that β is a supervertex supported on the base vertices $1, 2$.
Also add a nested-neighborhood attribute

$$\nu: \ \{1, 2, 3\} \to \mathcal{P}_f^2(\{1, 2, 3\})$$

given by

$$\nu(1) = \{\{2, 3\}, \{2\}\}, \qquad \nu(2) = \{\{1\}\}, \qquad \nu(3) = \{\{1, 2\}, \emptyset\}.$$

This is exactly a 2-nested neighborhood assignment of CNNG type on the base vertices.

(5) Weight/tensor layer (ATN flavor). Take $K = \mathbb{R}$. Define weights on R_2 and R_3 by

$$w_{R_2}(1,2) = 1, \qquad w_{R_2}(2,3) = 2, \qquad w_{R_3}(1,2,3) = 5.$$

Equivalently, one may regard these as the nonzero entries of a 2-tensor and a 3-tensor on the base vertices.

(6) Closure layer (CIG flavor). Let $Y = \{1,2,3\} \subseteq X_{*,\mathsf{X}}$, and define a closure operator

$$\mathrm{cl} : \mathcal{P}(Y) \to \mathcal{P}(Y)$$

generated by the rule

$$\{1,2\} \Rightarrow 3.$$

Concretely, $\mathrm{cl}(S)$ is the smallest subset of Y containing S and closed under the rule above. For example,

$$\mathrm{cl}(\{1\}) = \{1\}, \qquad \mathrm{cl}(\{1,2\}) = \{1,2,3\}.$$

(7) Operational signature (OIG flavor). Let Σ_{op} contain two symbols:

$$f : (*, \mathsf{Raw}) \to (*, \mathsf{Clean}), \qquad g : (*, \mathsf{Clean}), (*, \mathsf{Clean}) \to (*, \mathsf{Model}),$$

with the second operation symmetric in its two inputs. Interpret them by

$$[\![f]\!](r_1) = c_1, \qquad [\![f]\!](r_2) = c_2,$$

$$[\![g]\!](c_1, c_2) = m, \qquad [\![g]\!](c_2, c_1) = m.$$

Then the composite term

$$h(x,y) := g\big(f(x), f(y)\big)$$

is a well-typed operation term from $\mathsf{Raw} \times \mathsf{Raw}$ to Model.

(8) Conclusion. The tuple built above is a finite CHGS:

$$\mathfrak{H} = \big(I, S, X, \Sigma_{\mathrm{rel}}, \Sigma_{\mathrm{op}}, [\![-]\!], \mathcal{E}, \mathcal{A}, \mathcal{W}, \mathcal{C}\big),$$

with $\mathcal{E} = \emptyset$ in this concrete example. It simultaneously realizes:

- a supervertex/support mechanism (β with $\mathrm{supp}(\beta) = \{1,2\}$),
- hyperedge incidence (via Inc),
- pairwise and ternary interactions (via R_2, R_3),
- weighted higher-order adjacency (via w_{R_2}, w_{R_3}),
- closure implication (via cl),
- and a typed compositional workflow (via f, g).

Hence this example illustrates how CHGS unifies several previously separate higher-order formalisms within one typed framework.

Remark 6.1.5 (Recovering temporal and multilayer variants inside CHGS)**.** *To obtain temporal or multilayer models explicitly, one simply uses a nontrivial context set I, for example*

$$I = T, \qquad \textit{or} \qquad I = L,$$

$$\textit{or} \qquad I = T \times L,$$

and then assigns carriers $X_{i,s}$ and relations whose profiles involve one or multiple contexts. In this way, intralayer, interlayer, and time-stamped interactions become ordinary typed relations in the CHGS sense.

6.2 Morphisms, Representation, Redundancy, and Comparison for CHGS

For precision, we first fix a concrete (but flexible) version of a CHGS.

Definition 6.2.1 (Complete Higher-Graphic Structure (CHGS), typed form)**.** A *Complete Higher-Graphic Structure* (CHGS) is a tuple

$$\mathcal{H} = \Big(\mathcal{S},\ (X_s)_{s\in\mathcal{S}},\ \Sigma_{\text{rel}}, \Sigma_{\text{op}}, \Sigma_{\text{att}}, \Sigma_{\text{cl}}, \Sigma_{\text{w}},\ \text{Int}\Big),$$

where:

1. $\mathcal{S}$ is a finite set of *sorts* (types), and X_s is the carrier set of sort s.
2. Σ_{rel} is a finite set of relation symbols. Each $\rho \in \Sigma_{\text{rel}}$ has a *profile*
$$\text{prof}(\rho) = (s_1, \dots, s_m) \in \mathcal{S}^m,$$
and is interpreted as
$$\text{Int}_\rho \subseteq X_{s_1} \times \cdots \times X_{s_m}.$$
3. Σ_{op} is a finite set of (possibly partial) operation symbols. Each $\omega \in \Sigma_{\text{op}}$ has a profile
$$\text{prof}(\omega) = (s_1, \dots, s_m; s) \in \mathcal{S}^{m+1},$$
and is interpreted as a partial map
$$\text{Int}_\omega : X_{s_1} \times \cdots \times X_{s_m} \rightharpoonup X_s.$$
4. Σ_{att} is a finite set of attribute symbols. Each $a \in \Sigma_{\text{att}}$ has a profile $(s; D_a)$, where D_a is a fixed attribute domain, and is interpreted as
$$\text{Int}_a : X_s \to D_a.$$
5. Σ_{cl} is a finite set of closure symbols. Each $c \in \Sigma_{\text{cl}}$ has a profile $(s_1, \dots, s_m)$ and is interpreted as a closure operator
$$\text{Int}_c : \mathcal{P}(X_{s_1} \times \cdots \times X_{s_m}) \to \mathcal{P}(X_{s_1} \times \cdots \times X_{s_m}),$$
satisfying extensivity, monotonicity, and idempotence.
6. Σ_{w} is a finite set of weight symbols. Each $w \in \Sigma_{\text{w}}$ is attached to a relation or operation symbol and takes values in a fixed codomain K_w (e.g. $\mathbb{R}$, $\mathbb{N}_0$, or a semiring), via a map
$$\text{Int}_w : \text{Dom}(w) \to K_w.$$

Remark 6.2.2. *The tuple above may be viewed as a layered object:*

$$\textit{(Sort layer)} + \textit{(Relational layer)} + \textit{(Operational layer)} +$$

$$\textit{(Attribute layer)} + \textit{(Closure layer)} + \textit{(Weight layer)}.$$

Different higher-order network formalisms activate different subsets of these layers.

Definition 6.2.3 (CHGS morphism)**.** Let $\mathcal{H}$ and $\mathcal{H}'$ be CHGSs over the *same typed signature*

$$(\mathcal{S}, \Sigma_{\text{rel}}, \Sigma_{\text{op}}, \Sigma_{\text{att}}, \Sigma_{\text{cl}}, \Sigma_{\text{w}})$$

and the same attribute/weight codomains. A *CHGS morphism*

$$\Phi : \mathcal{H} \to \mathcal{H}'$$

is a family of maps

$$\Phi = (\phi_s)_{s\in\mathcal{S}}, \qquad \phi_s : X_s \to X'_s,$$

such that the following hold.

1. **Relation preservation:** for every $\rho \in \Sigma_{\mathrm{rel}}$ with profile $(s_1, \ldots, s_m)$,
$$(x_1, \ldots, x_m) \in \mathrm{Int}_\rho \quad \Longrightarrow \quad (\phi_{s_1}(x_1), \ldots, \phi_{s_m}(x_m)) \in \mathrm{Int}'_\rho.$$
2. **Operation preservation:** for every $\omega \in \Sigma_{\mathrm{op}}$ with profile $(s_1, \ldots, s_m; s)$, whenever $\mathrm{Int}_\omega(x_1, \ldots, x_m)$ is defined,
$$\mathrm{Int}'_\omega\big(\phi_{s_1}(x_1), \ldots, \phi_{s_m}(x_m)\big) \text{ is defined, and}$$
$$\phi_s\big(\mathrm{Int}_\omega(x_1, \ldots, x_m)\big) = \mathrm{Int}'_\omega\big(\phi_{s_1}(x_1), \ldots, \phi_{s_m}(x_m)\big).$$
3. **Attribute preservation:** for every $a \in \Sigma_{\mathrm{att}}$ of profile $(s; D_a)$,
$$\mathrm{Int}_a(x) = \mathrm{Int}'_a(\phi_s(x)) \qquad (x \in X_s).$$
4. **Closure continuity:** for every $c \in \Sigma_{\mathrm{cl}}$ of profile $(s_1, \ldots, s_m)$, writing
$$\phi_{(s_1,\ldots,s_m)} = \phi_{s_1} \times \cdots \times \phi_{s_m},$$
we require for all $A \subseteq X_{s_1} \times \cdots \times X_{s_m}$,
$$\phi_{(s_1,\ldots,s_m)}\big(\mathrm{Int}_c(A)\big) \subseteq \mathrm{Int}'_c\big(\phi_{(s_1,\ldots,s_m)}(A)\big).$$
5. **Weight preservation:** for every $w \in \Sigma_{\mathrm{w}}$, if w is attached to a relation tuple or operation instance u,
$$\mathrm{Int}_w(u) = \mathrm{Int}'_w\big(\Phi_*(u)\big),$$
where $\Phi_*(u)$ is the tuple/instance obtained by applying the relevant ϕ_s componentwise.

Definition 6.2.4 (Strong embedding and CHGS isomorphism)**.** A CHGS morphism $\Phi : \mathcal{H} \to \mathcal{H}'$ is a *strong embedding* if each ϕ_s is injective and:

1. relation preservation is reflected (i.e. “if and only if” on the image),
2. operation definedness is reflected and operations commute exactly on the image,
3. closure preservation is exact on images:
$$\phi_{(s_1,\ldots,s_m)}\big(\mathrm{Int}_c(A)\big) = \mathrm{Int}'_c\big(\phi_{(s_1,\ldots,s_m)}(A)\big) \cap \mathrm{Im}(\phi_{(s_1,\ldots,s_m)}),$$
4. weights and attributes are reflected as well as preserved on the image.

A bijective strong embedding is called a *CHGS isomorphism*.

Proposition 6.2.5 (CHGSs form a category)**.** *For a fixed typed signature, CHGSs and CHGS morphisms form a category* **CHGS***.*

Proof. Let $\mathcal{H}$ be a CHGS. The family of identity maps
$$\mathrm{id}_{\mathcal{H}} = (\mathrm{id}_{X_s})_{s \in \mathcal{S}}$$
clearly preserves relations, operations, attributes, closures, and weights, so it is a CHGS morphism.

Now let
$$\Phi = (\phi_s)_{s \in \mathcal{S}} : \mathcal{H} \to \mathcal{H}', \qquad \Psi = (\psi_s)_{s \in \mathcal{S}} : \mathcal{H}' \to \mathcal{H}''$$
be CHGS morphisms. Define the componentwise composite
$$\Psi \circ \Phi = (\psi_s \circ \phi_s)_{s \in \mathcal{S}}.$$
Each preservation condition is stable under composition:

- relation preservation follows by two successive applications of the relation-preservation implication;
- operation preservation follows from commutativity of each operation square and composition of partial maps;
- attribute preservation is immediate by substitution;

- closure continuity follows from

$$(\psi \times \cdots \times \psi)\big(\Phi(\mathrm{cl}(A))\big) \subseteq (\psi \times \cdots \times \psi)\big(\mathrm{cl}'(\Phi(A))\big) \subseteq \mathrm{cl}''\big((\Psi \circ \Phi)(A)\big);$$

- weight preservation is also immediate by substitution.

Associativity holds because function composition is associative componentwise. Hence the axioms of a category are satisfied. □

Definition 6.2.6 (Admissible higher-order model)**.** A higher-order model $\mathcal{M}$ is called *CHGS-admissible* if its primitive data can be presented by:

1. a finite family of carrier sets $(Y_t)_{t \in T}$;
2. finitely many typed relations on finite products of the Y_t;
3. finitely many typed (possibly partial) operations between finite products of the Y_t;
4. optionally, typed attributes, closure operators on typed products, and weight maps.

Any axioms of $\mathcal{M}$ (e.g. symmetry, associativity, simplicial closure, operad axioms, monoidal coherence constraints, etc.) are imposed as first-order or algebraic conditions on these interpreted symbols.

Theorem 6.2.7 (CHGS representation theorem)**.** *Every CHGS-admissible higher-order model $\mathcal{M}$ admits a faithful CHGS representation. More precisely, there exists a CHGS $\mathcal{H}_{\mathcal{M}}$ and an injective map of underlying data*

$$\iota : \mathcal{M} \hookrightarrow \mathcal{H}_{\mathcal{M}}$$

such that ι extends to a strong embedding of $\mathcal{M}$ into the reduct of $\mathcal{H}_{\mathcal{M}}$ generated by the symbols corresponding to the primitives of $\mathcal{M}$. In particular, $\mathcal{M}$ is isomorphic to a CHGS-substructure of $\mathcal{H}_{\mathcal{M}}$.

Proof. Let $\mathcal{M}$ be CHGS-admissible. By Definition 6.2.6, its primitives consist of:

- carrier sets $(Y_t)_{t \in T}$,
- typed relations $(R_i)_{i \in I}$,
- typed partial operations $(f_j)_{j \in J}$,
- and optionally attributes, closures, and weights.

Construct a typed signature as follows:

$$\mathcal{S} := T.$$

For each primitive relation $R_i \subseteq Y_{t_1} \times \cdots \times Y_{t_m}$, add a relation symbol ρ_i of profile $(t_1, \ldots, t_m)$. For each primitive partial operation $f_j : Y_{u_1} \times \cdots \times Y_{u_\ell} \rightharpoonup Y_v$, add an operation symbol ω_j of profile $(u_1, \ldots, u_\ell; v)$. Similarly, add attribute, closure, and weight symbols corresponding to the optional primitives.

Now define a CHGS $\mathcal{H}_{\mathcal{M}}$ by setting:

$$X_t := Y_t \qquad (t \in T),$$

and interpreting each symbol by the corresponding primitive object in $\mathcal{M}$:

$$\mathrm{Int}_{\rho_i} := R_i, \qquad \mathrm{Int}_{\omega_j} := f_j,$$

and analogously for attributes, closures, and weights.

By construction, all primitive data of $\mathcal{M}$ are represented exactly inside $\mathcal{H}_{\mathcal{M}}$, and any axioms of $\mathcal{M}$ become properties of this interpretation (they are not lost, only re-expressed in the CHGS signature). The identity maps on each carrier Y_t define a strong embedding of $\mathcal{M}$ into the appropriate reduct of $\mathcal{H}_{\mathcal{M}}$. Faithfulness follows because the interpretation is literal: distinct primitive relations/operations remain distinct symbols (or distinct interpretations) in the CHGS. □

Corollary 6.2.8 (Embedding conditions for common models)**.** *Each of the following admits a CHGS representation once its primitives are encoded as typed sets/relations/operations:*

1. *hypergraphs, multihypergraphs, superhypergraphs, recursive/nested/hierarchical variants;*
2. *relational-arity graphs (RAGs);*

3. *adjacency-tensor networks (ATNs), by using typed k-ary relations with weights;*
4. *closure-implication graphs (CIGs), by using a closure operator in the closure layer;*
5. *coalgebraic nested-neighborhood graphs (CNNGs), by adding sorts for finite powerset iterates and a typed map γ;*
6. *operadic interaction graphs (OIGs) and symmetric monoidal wiring graphs (SMWGs), by taking sorts for colors/objects/morphisms (and, if needed, operation objects) and interpreting composition/substitution/typing as operations and relations.*

Proof. Each listed model is CHGS-admissible under the indicated encoding. Apply Theorem 6.2.7. □

Definition 6.2.9 (Layer reduct)**.** Let

$$\Lambda = \{\mathrm{Sort}, \mathrm{Rel}, \mathrm{Op}, \mathrm{Att}, \mathrm{Cl}, \mathrm{W}\}$$

be the set of CHGS layers (with the Sort layer always retained). For a CHGS $\mathcal{H}$ and $L \subseteq \Lambda$ with Sort $\in L$, the *layer reduct*

$$\mathcal{H}\restriction_L$$

is obtained by forgetting all symbols and interpretations whose layers are not in L.

Definition 6.2.10 (Redundant and essential layers)**.** Let $\mathcal{H}$ be a CHGS and let $\lambda \in \Lambda \setminus \{\mathrm{Sort}\}$. We say that λ is *redundant in* $\mathcal{H}$ if there exists a reconstruction procedure

$$\mathcal{R}_\lambda$$

(which is uniform on the fixed signature class) such that

$$\mathcal{R}_\lambda(\mathcal{H}\restriction_{\Lambda\setminus\{\lambda\}}) \cong \mathcal{H}$$

as CHGSs. Otherwise, λ is called *essential*.

Proposition 6.2.11 (Redundancy elimination preserves semantics)**.** *If a layer λ is redundant in $\mathcal{H}$, then removing λ does not lose information up to CHGS isomorphism:*

$$\mathcal{H} \cong \mathcal{R}_\lambda(\mathcal{H}\restriction_{\Lambda\setminus\{\lambda\}}).$$

In particular, any invariant or query that is preserved under CHGS isomorphism has the same value on both sides.

Proof. This is immediate from Definition 6.2.10. By assumption, the reconstruction yields a CHGS isomorphic to $\mathcal{H}$. Isomorphism-invariant quantities and properties therefore coincide. □

Theorem 6.2.12 (Existence of a minimal-by-inclusion CHGS core)**.** *Let $\mathcal{H}$ be a CHGS. Then there exists a subset $L_* \subseteq \Lambda$ with* Sort $\in L_*$ *such that:*

1. *$\mathcal{H}\restriction_{L_*}$ has no redundant layer (relative to L_*),*
2. *$\mathcal{H}$ can be recovered from $\mathcal{H}\restriction_{L_*}$ by iterated reconstruction of redundant layers,*
3. *L_* is minimal by inclusion among subsets of layers with this property.*

Such a core need not be unique.

Proof. Start with $L_0 := \Lambda$. If L_i contains a redundant layer $\lambda_i \neq$ Sort, remove it and set

$$L_{i+1} := L_i \setminus \{\lambda_i\}.$$

Because Λ is finite, this process terminates after finitely many steps at a set L_* containing no redundant layer. By repeated application of Proposition 6.2.11, the information lost at each step can be reconstructed, so $\mathcal{H}$ is recoverable from $\mathcal{H}\restriction_{L_*}$.

Minimality by inclusion holds by construction: if some layer in L_* were removable while retaining recoverability, it would be redundant relative to L_*, contradicting termination.

Nonuniqueness may occur because different choices of removable layers at intermediate steps may lead to different minimal subsets. □

Proposition 6.2.13 (A practical criterion for essentiality)**.** *Fix a layer* $\lambda \in \Lambda \setminus \{\mathrm{Sort}\}$*. If there exist two CHGSs* $\mathcal{H}_1, \mathcal{H}_2$ *on the same carriers and same non-*λ *layers such that*

$$\mathcal{H}_1 \restriction_{\Lambda\setminus\{\lambda\}} = \mathcal{H}_2 \restriction_{\Lambda\setminus\{\lambda\}}, \qquad \mathcal{H}_1 \not\cong \mathcal{H}_2,$$

then λ *is essential (for that signature class).*

Proof. If λ were redundant, then both $\mathcal{H}_1$ and $\mathcal{H}_2$ would be reconstructed (up to isomorphism) from the same reduct

$$\mathcal{H}_1 \restriction_{\Lambda\setminus\{\lambda\}} = \mathcal{H}_2 \restriction_{\Lambda\setminus\{\lambda\}},$$

forcing $\mathcal{H}_1 \cong \mathcal{H}_2$, a contradiction. □

Definition 6.2.14 (CHGS comparison profile)**.** Let $\mathcal{H}$ be a *finite* CHGS (all carriers finite). We define the following comparison indices.

(i) Expressiveness profile. Define the vector

$$\mathrm{Expr}(\mathcal{H}) = \Big(|\mathcal{S}|,\ a^{\mathrm{rel}}_{\max},\ a^{\mathrm{op}}_{\max},\ |\Sigma_{\mathrm{rel}}|,\ |\Sigma_{\mathrm{op}}|,\ |\Sigma_{\mathrm{cl}}|,\ |\Sigma_{\mathrm{w}}|\Big),$$

where

$$a^{\mathrm{rel}}_{\max} = \max_{\rho\in\Sigma_{\mathrm{rel}}} \mathrm{arity}(\rho), \qquad a^{\mathrm{op}}_{\max} = \max_{\omega\in\Sigma_{\mathrm{op}}} \mathrm{arity}_{\mathrm{in}}(\omega).$$

This records how many typed entities, interaction arities, closure mechanisms, and weighted components the model can express natively.

(ii) Compositionality profile. Let $G = (G_s)_{s\in\mathcal{S}}$ be a chosen family of *primitive generators* with $G_s \subseteq X_s$. Define $\mathrm{CompCl}_{\mathcal{H}}(G)$ as the least family $Y = (Y_s)_{s\in\mathcal{S}}$ such that:

1. $G_s \subseteq Y_s$ for all s,
2. Y is closed under all interpreted operations Int_ω,
3. for each closure symbol c, the relevant interpreted relation sets generated from Y are closed under Int_c.

The *compositional coverage* is

$$\mathrm{Comp}_G(\mathcal{H}) = \frac{\sum_{s\in\mathcal{S}} |Y_s|}{\sum_{s\in\mathcal{S}} |X_s|}, \qquad Y = \mathrm{CompCl}_{\mathcal{H}}(G).$$

A higher value indicates that a larger proportion of the CHGS is generated compositionally from a small primitive core.

(iii) Learnability proxy (description-length type). Define the *description-length proxy*

$$\mathrm{LearnDL}(\mathcal{H}) = \sum_{s\in\mathcal{S}} \lceil \log_2(|X_s|+1)\rceil \ +\ \sum_{\rho\in\Sigma_{\mathrm{rel}}} \big|\mathrm{Int}_\rho\big| \ +\ \sum_{\omega\in\Sigma_{\mathrm{op}}} \big|\mathrm{graph}(\mathrm{Int}_\omega)\big| \ +\ \sum_{w\in\Sigma_{\mathrm{w}}} \big|\mathrm{dom}(\mathrm{Int}_w)\big|.$$

Here $\mathrm{graph}(\mathrm{Int}_\omega)$ is the graph of the partial function Int_ω. Smaller LearnDL is interpreted as a lower-complexity proxy and typically indicates easier memorization/learning of the instance.

Remark 6.2.15. *The three indices above are intended as* comparison profiles*, not universal absolute measures. They may be refined (e.g. weighted versions, MDL variants, statistical sample-complexity surrogates, or task-specific expressiveness scores) depending on the application.*

Proposition 6.2.16 (Isomorphism invariance of the comparison profile)**.** *If* $\mathcal{H} \cong \mathcal{H}'$ *as CHGSs, then:*

1. $\mathrm{Expr}(\mathcal{H}) = \mathrm{Expr}(\mathcal{H}')$,
2. *for every generator family* G*, the corresponding image generator family* $G' = \Phi(G)$ *satisfies*

$$\mathrm{Comp}_G(\mathcal{H}) = \mathrm{Comp}_{G'}(\mathcal{H}'),$$

3. $\mathrm{LearnDL}(\mathcal{H}) = \mathrm{LearnDL}(\mathcal{H}')$.

Proof. Let $\Phi = (\phi_s)_{s\in\mathcal{S}} : \mathcal{H} \to \mathcal{H}'$ be a CHGS isomorphism. By definition, each ϕ_s is bijective and preserves all interpreted relations, operations, closures, and weights exactly.

1. The signature is the same, so $|\mathcal{S}|$, the numbers of symbols, and arities coincide.
2. Because Φ commutes with all operations and closures, the compositional closure of G maps bijectively onto the compositional closure of $G' = \Phi(G)$. Hence the numerator and denominator in Comp_G are preserved.
3. Since Φ is bijective and preserves all interpretations exactly, the cardinalities of each interpreted relation, operation graph, and weight domain are preserved, as are the carrier cardinalities. Therefore LearnDL is unchanged.

□

Proposition 6.2.17 (Monotonicity of expressiveness under signature extension)**.** *Let $\mathcal{H}$ and $\widetilde{\mathcal{H}}$ be CHGSs on the same carriers, where $\widetilde{\mathcal{H}}$ is obtained from $\mathcal{H}$
by adding new relation/operation/closure/weight symbols (with interpretations) and keeping all old symbols unchanged. Then, componentwise,*

$$\mathrm{Expr}(\mathcal{H}) \leq \mathrm{Expr}(\widetilde{\mathcal{H}}),$$

where the inequality is interpreted coordinatewise.

Proof. Adding symbols cannot decrease the number of sorts or symbols; it also cannot decrease maximum arities because all previous symbols remain present. Hence each coordinate is nondecreasing. □

Proposition 6.2.18 (Redundancy reduction and learnability proxy)**.** *Let $\mathcal{H}$ be a finite CHGS and suppose a layer λ is redundant. If the reconstruction of λ from the reduct is deterministic and not stored explicitly in the reduct, then the reduced representation*

$$\mathcal{H}\restriction_{\Lambda\setminus\{\lambda\}}$$

has

$$\mathrm{LearnDL}\big(\mathcal{H}\restriction_{\Lambda\setminus\{\lambda\}}\big) \leq \mathrm{LearnDL}(\mathcal{H}),$$

with strict inequality whenever λ contributes at least one explicitly stored interpreted symbol/instance.

Proof. By removing a redundant layer, one deletes explicit interpreted symbols and/or stored interpreted data from the description. Under the stated convention (the layer is reconstructed, not stored), no new explicit data are added. Therefore the sum defining LearnDL cannot increase. It decreases strictly if at least one nonempty interpreted component from λ is removed from explicit storage. □

Proposition 6.2.19 (Expressiveness as an embedding preorder)**.** *Define*

$$\mathcal{H}_1 \preccurlyeq_{\mathrm{emb}} \mathcal{H}_2$$

if there exists a strong embedding $\mathcal{H}_1 \hookrightarrow \mathcal{H}_2$ (possibly after passing to a signature-preserving reduct of $\mathcal{H}_2$). Then $\preccurlyeq_{\mathrm{emb}}$ is a preorder on CHGSs.

Proof. Reflexivity holds via the identity isomorphism. Transitivity holds because the composition of strong embeddings is again a strong embedding (this is checked componentwise exactly as in Proposition 6.2.5, with injectivity and reflection properties preserved under composition). □

Remark 6.2.20 (Interpretation of the indices in practice)**.** *In applications, one often compares CHGS instances (or CHGS encodings of models) using all four viewpoints together:*

- Embedding preorder $\preccurlyeq_{\mathrm{emb}}$*: qualitative expressiveness,*
- $\mathrm{Expr}(\mathcal{H})$*: native structural capacity,*
- $\mathrm{Comp}_G(\mathcal{H})$*: degree of compositional generation from primitives,*
- $\mathrm{LearnDL}(\mathcal{H})$*: complexity proxy related to learnability/compressibility.*

This yields a mathematically explicit comparison framework for discussing trade-offs among expressiveness, compositionality, and practical learnability.

7 Conclusion

This book presented a comprehensive overview of mathematical notions that can be used to model higher-order networks. It surveyed foundational concepts, extensional frameworks, and newly introduced formalisms, with an emphasis on their structural principles, relationships, and modeling roles.

We expect that future research will further develop these concepts through uncertainty-aware extensions based on Fuzzy Graphs[296, 46], Intuitionistic Fuzzy Graphs[297, 298], Neutrosophic Graphs[299, 300, 301], and Plithogenic Graphs[302, 303]. In particular, it is natural to study how membership/non-membership (and indeterminacy) degrees can be lifted from vertices and edges to higher-order objects such as hyperedges, simplices, supervertices, and even meta-nodes in iterated constructions, while preserving basic consistency axioms and functorial behavior under morphisms.

From a computational perspective, promising directions include designing scalable algorithms for (i) higher-order centrality and community detection, (ii) influence/flow propagation on multiway interactions, and (iii) learning-based inference on higher-order data (e.g., message passing on factor/Tanner-style representations, or tensor-based contractions for structured multiway dependencies). We also anticipate broader applications in decision-support systems (group evaluation and consensus under multi-criteria constraints), machine learning (hypergraph and simplicial neural models), and other domains where complex higher-order interactions must be modeled and analyzed.

Disclaimer

Funding

This study did not receive any financial or external support from organizations or individuals.

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 book. Finally, we are grateful to all those who supported us in various ways during this project.

Data Availability

This research is purely theoretical, involving no data collection or analysis. We encourage future researchers to pursue empirical investigations to further develop and validate the concepts introduced here.

Ethical Approval

As this research is entirely theoretical in nature and does not involve human participants or animal subjects, no ethical approval is required.

Use of Generative AI and AI-Assisted Tools

We use generative AI and AI-assisted tools only for limited tasks such as English grammar checking, and we do not use them in any way that violates ethical standards.

Conflicts of Interest

The authors confirm that there are no conflicts of interest related to the research or its publication.

Disclaimer

This work presents theoretical concepts that have not yet undergone practical testing or validation. Future researchers are encouraged to apply and assess these ideas in empirical contexts. While every effort has been made to ensure accuracy and appropriate referencing, unintentional errors or omissions may still exist. Readers are advised to verify referenced materials on their own. The views and conclusions expressed here are the authors' own and do not necessarily reflect those of their affiliated organizations.

Appendix (List of Tables)

*

Appendix (List of Figures)

*

Bibliography

[1] Reinhard Diestel. *Graph theory.* Springer (print edition); Reinhard Diestel (eBooks), 2024.

[2] Jonathan L Gross, Jay Yellen, and Mark Anderson. *Graph theory and its applications.* Chapman and Hall/CRC, 2018.

[3] Alain Bretto. Hypergraph theory. *An introduction. Mathematical Engineering. Cham: Springer*, 1, 2013.

[4] Florentin Smarandache. *Extension of HyperGraph to n-SuperHyperGraph and to Plithogenic n-SuperHyperGraph, and Extension of HyperAlgebra to n-ary (Classical-/Neutro-/Anti-) HyperAlgebra.* Infinite Study, 2020.

[5] T. Fujita. Metahypergraphs, metasuperhypergdynaraphs, and iterated metagraphs: Modeling graphs of graphs, hypergraphs of hypergraphs, superhypergraphs of superhypergraphs, and beyond. *Current Research in Interdisciplinary Studies*, 4(5):1–23, 2025.

[6] Christian Bick, Elizabeth Gross, Heather A Harrington, and Michael T Schaub. What are higher-order networks? *SIAM review*, 65(3):686–731, 2023.

[7] Soumen Majhi, Matjaž Perc, and Dibakar Ghosh. Dynamics on higher-order networks: A review. *Journal of the Royal Society Interface*, 19(188), 2022.

[8] Hao Tian and Reza Zafarani. Higher-order networks representation and learning: A survey. *ACM SIGKDD Explorations Newsletter*, 26(1):1–18, 2024.

[9] Shahin Mohammadi, David F Gleich, Tamara G Kolda, and Ananth Grama. Triangular alignment (tame): A tensor-based approach for higher-order network alignment. *IEEE/ACM transactions on computational biology and bioinformatics*, 14(6):1446–1458, 2016.

[10] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, pages 4602–4609, 2019.

[11] Joydeep Ghosh and Yoan Shin. Efficient higher-order neural networks for classification and function approximation. *International Journal of Neural Systems*, 3(04):323–350, 1992.

[12] Cuiping Ren, Bianbian Chen, Fengjie Xie, Xuan Zhao, Jiaqian Zhang, and Xueyan Zhou. Understanding hazardous materials transportation accidents based on higher-order network theory. *International journal of environmental research and public health*, 19(20):13337, 2022.

[13] Takaaki Fujita and Arif Mehmood. Actor hypernetworks and urban road hypernetworks with real-life applications. *Neutrosophic Computing and Machine Learning*, 41:143–171, 2025.

[14] Takaaki Fujita. River network hypergraphs and transportation network hypergraphs: A graph-theoretic approach for geoscientific and civil applications. *Interdisciplinary Studies on Applied Science*, 2(1):19–31, 2025.

[15] Fengjie Xie, Jiaxin Huang, Cuiping Ren, and Mengchan Wei. Research on bus route optimization based on a higher-order network model. In *Fourth International Conference on Smart City Engineering and Public Transportation (SCEPT 2024)*, volume 13160, pages 314–322. SPIE, 2024.

[16] Xuhui Lin, Long Chen, Qiuchen Lu, Pengjun Zhao, and Tao Cheng. Revealing higher-order interactions through multimodal irreversibility in flood-affected transportation networks. *Reliability Engineering & System Safety*, page 111726, 2025.

[17] Unai Alvarez-Rodriguez, Federico Battiston, Guilherme Ferraz de Arruda, Yamir Moreno, Matjaž Perc, and Vito Latora. Evolutionary dynamics of higher-order interactions in social networks. *Nature Human Behaviour*, 5(5):586–595, 2021.

[18] Giulia Cencetti, Federico Battiston, Bruno Lepri, and Márton Karsai. Temporal properties of higher-order interactions in social networks. *Scientific reports*, 11(1):7028, 2021.

[19] T. Fujita and F. Smarandache. Competition super-hypergraphs: Revealing hierarchical competition in real-world networks. *Journal of Algebra and Applied Mathematics*, 23(2):97–116, 2025.

[20] Takaaki Fujita. Cognitive hypergraphs and superhypergraphs: A novel framework for complex relational modeling. *Neutrosophic Computing & Machine Learning*, 39, 2025.

[21] Yong-Jian He, Xiao-Ke Xu, and Jing Xiao. Predicting higher order links in social interaction networks. *IEEE Transactions on Computational Social Systems*, 11(2):2796–2806, 2023.

[22] Hao Peng, Rui Zhang, Bo Zhang, Cheng Qian, Ming Zhong, Shenghong Li, Jianmin Han, Dandan Zhao, and Wei Wang. Prediction of social influence in higher-order networks. *Information Sciences*, page 123075, 2026.

[23] Thomas Gebhart and Russell James Funk. The emergence of higher-order structure in scientific and technological knowledge networks. In *Academy of Management Proceedings.* Academy of Management Briarcliff Manor, NY 10510, 2023.

[24] Yousra Moh Ousellam, Bikram Pratim Bhuyan, Rachida Fissoune, Galina Ivanova, and Amar Ramdane-Cherif. Learning directed knowledge using higher-ordered neural networks: Building a predictive framework. *Applied Sciences*, 15(20):11085, 2025.

[25] Thomas Gaudelet, Noel Malod-Dognin, and Nataša Pržulj. Higher-order molecular organization as a source of biological function. *Bioinformatics*, 34(17):i944–i953, 2018.

[26] Takaaki Fujita. Unifying grain boundary networks and crystal graphs: A hypergraph and superhypergraph perspective in material sciences. *Asian Journal of Advanced Research and Reports*, 19(5):344–379, 2025.

[27] Guillermo Restrepo. Higher order structures in chemistry: hypergraphs reshape the molecule and the reaction. *Digital Discovery*, 2026.

[28] Takaaki Fujita and Arkan A Ghaib. Toward a unified theory of brain hypergraphs and symptom hypernetworks in medicine and neuroscience. *Advances in Research*, 26(3):522–565, 2025.

[29] Bengier Ülgen Kilic. *Characterizing Dynamics on and of Networks via Higher-Order Interactions: Applications in Computational Neuroscience*. State University of New York at Buffalo, 2023.

[30] Fernando Pineda. Generalization of back propagation to recurrent and higher order neural networks. In *Neural information processing systems*, 1987.

[31] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, pages 3558–3565, 2019.

[32] Thomas Schnake, Oliver Eberle, Jonas Lederer, Shinichi Nakajima, Kristof T Schütt, Klaus-Robert Müller, and Grégoire Montavon. Higher-order explanations of graph neural networks via relevant walks. *IEEE transactions on pattern analysis and machine intelligence*, 44(11):7581–7596, 2021.

[33] C Lee Giles and Tom Maxwell. Learning, invariance, and generalization in high-order neural networks. *Applied optics*, 26(23):4972–4978, 1987.

[34] Yu Wang and Qilong Zhao. Multi-order hypergraph convolutional neural network for dynamic social recommendation system. *IEEE Access*, 10:87639–87649, 2022.

[35] Siyue Li, Tian Jin, Hao Luo, Erfan Wang, and Ranting Tao. Recommendation model based on higher-order semantics and node attention in heterogeneous graph neural networks. *Mathematics*, 13(9):1479, 2025.

[36] Darnbi Sakong, Thanh Trung Huynh, and Jun Jo. Hypergraph diffusion for high-order recommender systems. *arXiv preprint arXiv:2501.16722*, 2025.

[37] Zetian Shen, Hongjun Wang, Jiyuan Chen, and Xuan Song. Resilience inference for supply chains with hypergraph neural network. *arXiv preprint arXiv:2511.06208*, 2025.

[38] Takaaki Fujita. Telecommunications hypernetwork and telecommunications superhypernetwork. *Intelligence Modeling in Electromechanical Systems*, 2(1):16–31, 2025.

[39] Jules Hedges, Paulo Oliva, Evguenia Shprits, Viktor Winschel, and Philipp Zahn. Higher-order decision theory. In *International Conference on Algorithmic Decision Theory*, pages 241–254. Springer, 2017.

[40] Roshan Joy Martis, U Rajendra Acharya, KM Mandana, Ajoy Kumar Ray, and Chandan Chakraborty. Cardiac decision making using higher order spectra. *Biomedical Signal Processing and Control*, 8(2):193–203, 2013.

[41] Mohammad Hamidi, Florentin Smarandache, and Mohadeseh Taghinezhad. *Decision Making Based on Valued Fuzzy Superhypergraphs*. Infinite Study, 2023.

[42] William F Weiss IV, John P Gabrielson, Wasfi Al-Azzam, Guodong Chen, Darryl L Davis, Tapan K Das, David B Hayes, Damian Houde, and Satish K Singh. Technical decision making with higher order structure data: perspectives on higher order structure characterization from the biopharmaceutical industry. *Journal of Pharmaceutical Sciences*, 105(12):3465–3470, 2016.

[43] Mohammad Hamidi and Mohadeseh Taghinezhad. *Application of Superhypergraphs-Based Domination Number in Real World*. Infinite Study, 2023.

[44] Qian Wang and Zengtai Gong. An application of fuzzy hypergraphs and hypergraphs in granular computing. *Inf. Sci.*, 429:296–314, 2018.

[45] Xiang Ma, Tianlong Zhao, Qiang Guo, Xue mei Li, and Caiming Zhang. Fuzzy hypergraph network for recommending top-k profitable stocks. *Inf. Sci.*, 613:239–255, 2022.

[46] John N Mordeson and Premchand S Nair. *Fuzzy graphs and fuzzy hypergraphs*, volume 46. Physica, 2012.

[47] Anam Luqman, Muhammad Akram, and Florentin Smarandache. Complex neutrosophic hypergraphs: New social network models. *Algorithms*, 12:234, 2019.

[48] Muhammad Akram, Sundas Shahzadi, and AB Saeid. Single-valued neutrosophic hypergraphs. *TWMS Journal of Applied and Engineering Mathematics*, 8(1):122–135, 2018.

[49] Muhammad Akram and Anam Luqman. Intuitionistic single-valued neutrosophic hypergraphs. *OPSEARCH*, 54:799 – 815, 2017.

[50] Nivetha Martin and Florentin Smarandache. Concentric plithogenic hypergraph based on plithogenic hypersoft sets - a novel outlook. *Neutrosophic Sets and Systems*, 33:5, 2020.

[51] Ehab Roshdy, Marwa Khashaba, and Mariam Emad Ahmed Ali. Neutrosophic super-hypergraph fusion for proactive cyber-attack countermeasures: A soft computing framework. *Neutrosophic Sets and Systems*, 94:232–252, 2025.

[52] Takaaki Fujita and Florentin Smarandache. *Neutrosophic soft n-super-hypergraphs with real-world applications*. Infinite Study, 2025.

[53] Eduardo Martín Campoverde Valencia, Jessica Paola Chuisaca Vásquez, and Francisco Ángel Becerra Lois. 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*, 84:341–355, 2025.

[54] Julio Cesar Méndez Bravo, Claudia Jeaneth Bolanos Piedrahita, Manuel Alberto Méndez Bravo, and Luis Manuel Pilacuan-Bonete. Integrating smed and industry 4.0 to optimize processes with plithogenic n-superhypergraphs. *Neutrosophic Sets and Systems*, 84:328–340, 2025.

[55] Berrocal Villegas Salomón Marcos, Montalvo Fritas Willner, Berrocal Villegas Carmen Rosa, Flores Fuentes Rivera María Yissel, Espejo Rivera Roberto, Laura Daysi Bautista Puma, and Dante Manuel Macazana Fernández. Using plithogenic n-superhypergraphs to assess the degree of relationship between information skills and digital competencies. *Neutrosophic Sets and Systems*, 84:513–524, 2025.

[56] Muhammad Akram and Anam Luqman. A new decision-making method based on bipolar neutrosophic directed hypergraphs. *Journal of Applied Mathematics and Computing*, 57:547 – 575, 2017.

[57] Daniele Pretolani. Finding hypernetworks in directed hypergraphs. *European Journal of Operational Research*, 230(2):226–230, 2013.

[58] Takaaki Fujita. Review of some superhypergraph classes: Directed, bidirected, soft, and rough. *Advancing Uncertain Combinatorics through Graphization, Hyperization, and Uncertainization: Fuzzy, Neutrosophic, Soft, Rough, and Beyond (Second Volume)*, 2024.

[59] Takaaki Fujita. Directed superhypertree-width, mixed tree-width, and bidirected tree-width. *SuperHyperTopologies and SuperHyper Structures with their Applications*, page 114, 2025.

[60] Takaaki Fujita and Florentin Smarandache. Soft directed n-superhypergraphs with some real-world applications. *European Journal of Pure and Applied Mathematics*, 18(4):6643–6643, 2025.

[61] Lucas J Rusnak, Selena Li, Brian Xu, Eric Yan, and Shirley Zhu. Oriented hypergraphs: balanceability. *Discrete Mathematics*, 345(6):112832, 2022.

[62] Nathan Reff. Spectral properties of oriented hypergraphs. *arXiv preprint arXiv:1506.05054*, 2015.

[63] Ouail Kitouni and Nathan Reff. Lower bounds for the laplacian spectral radius of an oriented hypergraph. *Australas. J Comb.*, 74:408–422, 2019.

[64] Luke Duttweiler and Nathan Reff. Spectra of cycle and path families of oriented hypergraphs. *Linear Algebra and its Applications*, 578:251–271, 2019.

[65] Takaaki Fujita. Extensions of multidirected graphs: Fuzzy, neutrosophic, plithogenic, rough, soft, hypergraph, and superhypergraph variants. *International journal of topology*, 2(3):11, 2025.

[66] Takaaki Fujita and Florentin Smarandache. *HyperGraph and SuperHyperGraph Theory with Applications.* Neutrosophic Science International Association (NSIA) Publishing House, 2026.

[67] Florentin Smarandache. The cardinal of the m-powerset of a set of n elements used in the superhyperstructures and neutrosophic superhyperstructures. *Systems Assessment and Engineering Management*, 2:19–22, 2024.

[68] Claude Berge. *Hypergraphs: combinatorics of finite sets*, volume 45. Elsevier, 1984.

[69] Florentin Smarandache. *Introduction to the n-SuperHyperGraph-the most general form of graph today.* Infinite Study, 2022.

[70] Maissam Jdid, Florentin Smarandache, and Takaaki Fujita. A linear mathematical model of the vocational training problem in a company using neutrosophic logic, hyperfunctions, and superhyperfunction. *Neutrosophic Sets and Systems*, 87:1–11, 2025.

[71] Florentin Smarandache. *SuperHyperFunction, SuperHyperStructure, Neutrosophic SuperHyperFunction and Neutrosophic SuperHyperStructure: Current understanding and future directions.* Infinite Study, 2023.

[72] WB Vasantha Kandasamy, K Ilanthenral, and Florentin Smarandache. *Subset Vertex Multigraphs and Neutrosophic Multigraphs for Social Multi Networks.* Infinite Study, 2019.

[73] Brian Alspach and Cun-Quan Zhang. Cycle covers of cubic multigraphs. *Discrete mathematics*, 111(1-3):11–17, 1993.

[74] Landon Butler, Alejandro Parada-Mayorga, and Alejandro Ribeiro. Convolutional learning on multigraphs. *IEEE Transactions on Signal Processing*, 71:933–946, 2023.

[75] Siddhartha Sankar Biswas, Bashir Alam, and MN Doja. Intuitionistic fuzzy shortest path in a multigraph. In *International Conference on Recent Developments in Science, Engineering and Technology*, pages 533–540. Springer, 2017.

[76] Rajab Ali Borzooei and R Almallah. Inverse fuzzy multigraphs and planarity with application in decision-making. *Soft Computing*, 26(4):1531–1539, 2022.

[77] Anthony Shannon and Krassimir Atanassov. On intuitionistic fuzzy multigraphs and their index matrix interpretations. In *2004 2nd International IEEE Conference on'Intelligent Systems'. Proceedings (IEEE Cat. No. 04EX791)*, volume 2, pages 440–443. IEEE, 2004.

[78] Noga Alon. A simple algorithm for edge-coloring bipartite multigraphs. *Information Processing Letters*, 85(6):301–302, 2003.

[79] Fred Galvin. The list chromatic index of a bipartite multigraph. *Journal of Combinatorial Theory, Series B*, 63(1):153–158, 1995.

[80] Michael Tarsi. Decomposition of a complete multigraph into simple paths: nonbalanced handcuffed designs. *Journal of Combinatorial Theory, Series A*, 34(1):60–70, 1983.

[81] Zhihe Liang and Jinping Guo. Decomposition of complete multigraphs into crown graphs. *Journal of Applied Mathematics and Computing*, 32(2):507–517, 2010.

[82] Florentin Smarandache. *Nidus Idearum. Scilogs, III: viva la neutrosophia!*, volume 3. Infinite Study, 2017.

[83] M Baghernejad and Rajab Ali Borzooei. Results on soft graphs and soft multigraphs with application in controlling urban traffic flows. *Soft Computing*, 27(16):11155–11175, 2023.

[84] J Abderramán Marrero, Juan Núñez Valdés, and María Trinidad Villar. Associating hub-directed multigraphs to arrowhead matrices. *Mathematical Methods in the Applied Sciences*, 41(6):2360–2369, 2018.

[85] Donald Silberger. Eulerian directed multigraphs. *arXiv preprint arXiv:2408.12699*, 2024.

[86] Ray E Jennings and Yue Chen. Articular models for first-degree paraconsistent systems. In *9th IEEE International Conference on Cognitive Informatics (ICCI'10)*, pages 904–907. IEEE, 2010.

[87] Jia Shen. Packing and covering k-chain free subsets in boolean lattices. *Discrete mathematics*, 309(13):4628–4634, 2009.

[88] Melody Mae Cabigting Lunar and Renson Aguilar Robles. Characterization and structure of a power set graph. *International Journal of Advanced Research and Publications*, 3(6):1–4, 2019.

[89] Mark Ramras and Elizabeth Donovan. The automorphism group of a johnson graph. *SIAM Journal on Discrete Mathematics*, 25(1):267–270, 2011.

[90] Andrei Mikhailovich Raigorodskii and Petr Sergeevich Sinelnikov-Murylev. Johnson graphs, their random subgraphs, and some of their extremal characteristics. *Uspekhi Matematicheskikh Nauk*, 80(3):113–176, 2025.

[91] María Gracia Cornet, Tanja Dravec, and Pablo Torres. Domination in johnson graphs. *Procedia Computer Science*, 273:223–230, 2025.

[92] Louis Anthony Agong, Carmen Amarra, John S Caughman, Ari J Herman, and Taiyo S Terada. On the girth and diameter of generalized johnson graphs. *Discrete Mathematics*, 341(1):138–142, 2018.

[93] Zhecheng Yu, Liqiong Xu, Xuemin Wu, and Chuanye Zheng. On the super (edge)-connectivity of generalized johnson graphs. *International Journal of Foundations of Computer Science*, 35(05):579–593, 2024.

[94] Andrei Mikhailovich Raigorodskii and Petr Sergeevich Sinelnikov-Murylev. Johnson graphs, their random subgraphs, and some of their extremal characteristics. *Uspekhi Matematicheskikh Nauk*, 80(3):113–176, 2025.

[95] Ya-Chen Chen. Triangle-free hamiltonian kneser graphs. *Journal of Combinatorial Theory, Series B*, 89(1):1–16, 2003.

[96] Tristan Denley. The odd girth of the generalised kneser graph. *European Journal of Combinatorics*, 18(6):607–611, 1997.

[97] Hou Tin Chau, David Ellis, Ehud Friedgut, and Noam Lifshitz. On the maximum degree of induced subgraphs of the kneser graph. *arXiv preprint arXiv:2312.06370*, 2023.

[98] Torsten Mütze, Jerri Nummenpalo, and Bartosz Walczak. Sparse kneser graphs are hamiltonian. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 912–919, 2018.

[99] Noga Alon, Peter Frankl, and László Lovász. The chromatic number of kneser hypergraphs. *Transactions of the American Mathematical Society*, 298(1):359–370, 1986.

[100] Frédéric Meunier. The chromatic number of almost stable kneser hypergraphs. *Journal of Combinatorial Theory, Series A*, 118(6):1820–1828, 2011.

[101] S Morteza Mirafzal. The automorphism group of the bipartite kneser graph. *Proceedings-Mathematical Sciences*, 129(3):34, 2019.

[102] Torsten Mütze and Pascal Su. Bipartite kneser graphs are hamiltonian. *Combinatorica*, 37(6):1207–1219, 2017.

[103] Benjamin Braun. Independence complexes of stable kneser graphs. *arXiv preprint arXiv:0912.0720*, 2009.

[104] Carsten Schultz. The equivariant topology of stable kneser graphs. *Journal of Combinatorial Theory, Series A*, 118(8):2291–2318, 2011.

[105] Ke Liu, Mengyu Cao, and Mei Lu. Treewidth of the generalized kneser graphs. *arXiv preprint arXiv:2011.12725*, 2020.

[106] CA Rodger and Thomas Richard Whitt III. Path decompositions of kneser and generalized kneser graphs. *Canadian Mathematical Bulletin*, 58(3):610–619, 2015.

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

[108] Mathieu Poudret, Agnes Arnould, Jean-Paul Comet, and Pascale Le Gall. Graph transformation for topology modelling. In *International Conference on Graph Transformation*, pages 147–161. Springer, 2008.

[109] Erika Velazquez-Garcia, Ivan Lopez-Arevalo, and Victor Sosa-Sosa. Semantic graph-based approach for document organization. In *Distributed Computing and Artificial Intelligence: 9th International Conference*, pages 469–476. Springer, 2012.

[110] Runzhong Wang, Rui-Xi Wang, Mrunali Manjrekar, and Connor W Coley. Neural graph matching improves retrieval augmented generation in molecular machine learning. *arXiv preprint arXiv:2502.17874*, 2025.

[111] Ben Goertzel. Actpc-chem: Discrete active predictive coding for goal-guided algorithmic chemistry as a potential cognitive kernel for hyperon & primus-based agi. *arXiv preprint arXiv:2412.16547*, 2024.

[112] Jinze Wang, Jiong Jin, Lu Zhang, Hong-Ning Dai, Adriano Di-Pietro, and Tiehua Zhang. Towards spatial-temporal metahypergraph learning for multimodal few-shot fault diagnosis. *Journal of Industrial Information Integration*, page 100924, 2025.

[113] Takaaki Fujita. Note of nested superhypergraph and unified superhypergraph. *Annals of Pure and Applied Mathematics*, 33(1):5–13, 2026.

[114] Pengfei Huang and Minru Bai. An ideal-sparse generalized moment problem reformulation for completely positive tensor decomposition exploiting maximal cliques of multi-hypergraphs. *arXiv preprint arXiv:2505.15056*, 2025.

[115] Jiecheng Li, Xudong Luo, Guangquan Lu, and Shichao Zhang. Hyper-relational knowledge representation learning with multi-hypergraph disentanglement. In *Proceedings of the ACM on Web Conference 2025*, pages 3288–3299, 2025.

[116] Takaaki Fujita. Plithogenic line graph, star graph, and regular graph. *Plithogenic Logic and Computation*, 4:9–28, 2025.

[117] Ivan Gutman and Ernesto Estrada. Topological indices based on the line graph of the molecular graph. *Journal of chemical information and computer sciences*, 36(3):541–543, 1996.

[118] Muhammad Akram and Arooj Adeel. m-polar fuzzy graphs and m-polar fuzzy line graphs. *Journal of Discrete Mathematical Sciences and Cryptography*, 20:1597 – 1617, 2017.

[119] Ladislav Nebeskỳ. On the line graph of the square and the square of the line graph of a connected graph. *Časopis pro pěstování matematiky*, 98(3):285–287, 1973.

[120] Toto Nusantara, Desi Rahmadani, Yuliana Trisanti, Abdullah Bin Gani, et al. Anti fuzzy line graphs. In *Journal of Physics: Conference series*. IOP Publishing, 2021.

[121] Takaaki Fujita. Note for line and total superhypergraphs: Connecting vertices, edges, edges of edges, edges of edges of edges in hierarchical systems. *Journal of Mathematical Analysis and Modeling*, 6(3):11–43, 2025.

[122] Sumera Naz and Muhammad Aslam Malik. Single-valued neutrosophic line graphs. *TWMS Journal of Applied and Engineering Mathematics*, 8(2):483–494, 2018.

[123] Keneni Abera Tola, VN Srinivasa Rao Repalle, and Mamo Abebe Ashebo. Theory and application of interval-valued neutrosophic line graphs. *Journal of Mathematics*, 2024(1):5692756, 2024.

[124] Martin Knor and L'udovıt Niepel. Connectivity of iterated line graphs. *Discrete applied mathematics*, 125(2-3):255–266, 2003.

[125] M Knor, L Niepel, and L Šoltés. Centers in iterated line graphs. *Acta Math. Univ. Comenian*, 61:237–241, 1992.

[126] Stephen G Hartke and Aparna W Higgins. Maximum degree growth of the iterated line graph. *the electronic journal of combinatorics*, pages R28–R28, 1999.

[127] Yehong Shao. A bound on connectivity of iterated line graphs. *Electronic Journal of Graph Theory & Applications*, 10(2), 2022.

[128] Harishchandra S Ramane, Hanumappa B Walikar, Siddani Bhaskara Rao, B Devadas Acharya, Prabhakar Ramrao Hampiholi, Sudhir R Jog, and Ivan Gutman. Spectra and energies of iterated line graphs of regular graphs. *Applied mathematics letters*, 18(6):679–682, 2005.

[129] HS Ramane, DS Revankar, Ivan Gutman, and HB Walikar. Distance spectra and distance energies of iterated line graphs of regular graphs. *Publications de l'Institut Mathematique*, 85(99):39–46, 2009.

[130] Takaaki Fujita, Ajoy Kanti Das, and Suman Das. Molecular line HyperGraphs and SuperHyperGraphs. *Neutrosophic Knowledge*, 9:31–65, 2026.

[131] Eber Lenes, Exequiel Mallea-Zepeda, María Robbiano, and Jonnathan Rodríguez. On the diameter and incidence energy of iterated total graphs. *Symmetry*, 10(7):252, 2018.

[132] Gui-Xian Tian. The asymptotic behavior of (degree-) kirchhoff indices of iterated total graphs of regular graphs. *Discrete Applied Mathematics*, 233:224–230, 2017.

[133] ATHUL TB, ROY JOHN, AKHIL CK, MANJU VN, and SUBHA AB. The total graph of path related graphs and their extension. *Global & Stochastic Analysis*, 12(4), 2025.

[134] Mahipal Jadeja, Rahul Muthu, and Ravi Goyal. A new characterisation of total graphs. *Palestine Journal of Mathematics*, 13(4), 2024.

[135] Takaaki Fujita and Florentin Smarandache. *HyperGraph and SuperHyperGraph Theory with Applications (IV): Uncertain Graph Theory*, volume IV of *HyperGraph and SuperHyperGraph Theory with Applications*. Neutrosophic Science International Association (NSIA) Publishing House, 1.0 edition, 2026.

[136] Takaaki Fujita and Florentin Smarandache. *HyperGraph and SuperHyperGraph Theory with Applications (II): Graph Property and Parameter*, volume II of *HyperGraph and SuperHyperGraph Theory with Applications*. Neutrosophic Science International Association (NSIA) Publishing House, 1.0 edition, 2026.

[137] Naganand Yadati, RS Dayanidhi, S Vaishnavi, KM Indira, and G Srinidhi. Knowledge base question answering through recursive hypergraphs. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: main volume*, pages 448–454, 2021.

[138] Keli Long, Fangfa Fu, Defu Hu, Liangquan Qiao, Jinghan Zhou, and Jinxiang Wang. Automated cross-level verification for heterogeneous systems using attribute recursive hypergraph and fault-tolerant bus resolver. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2025.

[139] Tony Gracious and Ambedkar Dukkipati. Deep representation learning for forecasting recursive and multi-relational events in temporal networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

[140] Naganand Yadati. Neural message passing for multi-relational ordered and recursive hypergraphs. *Advances in Neural Information Processing Systems*, 33:3275–3289, 2020.

[141] Takaaki Fujita and Florentin Smarandache. *Representing Higher-Order Networks: A Survey of Graph-Based Frameworks*. Neutrosophic Science International Association (NSIA) Publishing House, 2026.

[142] Takaaki Fujita and Florentin Smarandache. *Theoretical Foundations of Superhypergraph and Plithogenic Graph Neural Networks*. Neutrosophic Science International Association (NSIA) Publishing House, 2026.

[143] T. Fujita and Florentin Smarandache. *Hyperstructures and Superhyperstructures: Theory and Selected Applications*. Neutrosophic Science International Association (NSIA) Publishing House, 2026.

[144] Takaaki Fujita. Tree-vertex graph: New hierarcal graph class, 2025.

[145] Danylo Lykov and Yuri Alexeev. Importance of diagonal gates in tensor network simulations. In *2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 447–452. IEEE, 2021.

[146] Yang Gao, Huanchen Zhai, Johnnie Gray, Ruojing Peng, Gunhee Park, Wen-Yuan Liu, Eirik F Kjønstad, and Garnet Kin-Lic Chan. Fermionic tensor network contraction for arbitrary geometries. *Physical Review Research*, 7(2):023193, 2025.

[147] Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[148] Georgii S Novikov, Maxim E Panov, and Ivan V Oseledets. Tensor-train density estimation. In *Uncertainty in artificial intelligence*, pages 1321–1331. PMLR, 2021.

[149] Zhen Qin and Zhihui Zhu. Computational and statistical guarantees for tensor-on-tensor regression with tensor train decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[150] Song Cheng, Lei Wang, Tao Xiang, and Pan Zhang. Tree tensor networks for generative modeling. *Physical Review B*, 99(15):155131, 2019.

[151] Gianluca Ceruti, Christian Lubich, and Hanna Walach. Time integration of tree tensor networks. *SIAM Journal on Numerical Analysis*, 59(1):289–313, 2021.

[152] Toshiya Hikihara, Hiroshi Ueda, Kouichi Okunishi, Kenji Harada, and Tomotoshi Nishino. Automatic structural optimization of tree tensor networks. *Physical Review Research*, 5(1):013031, 2023.

[153] Julian Schuhmacher, Marco Ballarin, Alberto Baiardi, Giuseppe Magnifico, Francesco Tacchino, Simone Montangero, and Ivano Tavernelli. Hybrid tree tensor networks for quantum simulation. *PRX Quantum*, 6(1):010320, 2025.

[154] Bertrand Michel and Anthony Nouy. Learning with tree tensor networks: complexity estimates and model selection. *Bernoulli*, 28(2):910–936, 2022.

[155] Domenico Pomarico. Multiscale entanglement renormalization ansatz: causality and error correction. *Dynamics*, 3(3):622–635, 2023.

[156] Philippe Corboz and Guifré Vidal. Fermionic multiscale entanglement renormalization ansatz. *Physical Review B—Condensed Matter and Materials Physics*, 80(16):165129, 2009.

[157] Vittorio Giovannetti, Simone Montangero, and Rosario Fazio. Quantum multiscale entanglement renormalization ansatz channels. *Physical review letters*, 101(18):180503, 2008.

[158] Glen Evenbly and Guifre Vidal. Tensor network renormalization yields the multiscale entanglement renormalization ansatz. *Physical review letters*, 115(20):200401, 2015.

[159] Didier Poilblanc and Norbert Schuch. Simplex z 2 spin liquids on the kagome lattice with projected entangled pair states: Spinon and vison coherence lengths, topological entropy, and gapless edge modes. *Physical Review B*, 87:140407, 2013.

[160] Zhi-Yuan Xie, Jack Chen, JF Yu, X Kong, Bruce Normand, and Tao Xiang. Tensor renormalization of quantum many-body systems using projected entangled simplex states. *Physical Review X*, 4(1):011025, 2014.

[161] Gulay Oguz and Bijan Davvaz. Soft topological hyperstructure. *J. Intell. Fuzzy Syst.*, 40:8755–8764, 2021.

[162] Sabahat A. Khan, Mohammad Y. Abbasi, Kostaq Hila, and Aakif F. Talee. A study of soft hyperideals in right regular la-semihypergroups. *Asian-European Journal of Mathematics*, 2023.

[163] Mehmet Ali YÜZBAŞIOĞLU. Implementing neutrosophic superhyperstructure in accounting information systems: A practical framework. *SuperHyperTopologies and SuperHyperStructures with their Applications: Collected Papers*, page 323, 2025.

[164] Hacı Mustafa PAKSOY, Mehmet Ali YÜZBAŞIOĞLU, and Yunus Emre. A theoretical framework of neutrosophic superhyperstructure in cost accounting: A new perspective for academic researchers. *SuperHyperTopologies and SuperHyperStructures with their Applications*, page 300, 2025.

[165] Poornima Ramakrishnan and Gayathri Palanisamy. Neutrosophic superhyperstructures: Foundations, transformations and applications. *SuperHyperTopologies and SuperHyperStructures with their Applications*, page 71, 2025.

[166] Yunus Emre KAHRAMAN, Hacı Mustafa PAKSOY, Murat KARAHAN, and Mehmet Ali YÜZBAŞIOĞLU. Implementing neutrosophic superhyperstructure in accounting information systems: A practical framework for financial decision making. *SuperHyperTopologies and SuperHyperStructures with their Applications*, page 323, 2025.

[167] M Al Tahan and Bijan Davvaz. Weak chemical hyperstructures associated to electrochemical cells. *Iranian Journal of Mathematical Chemistry*, 9(1):65–75, 2018.

[168] Adel Al-Odhari. Neutrosophic power-set and neutrosophic hyper-structure of neutrosophic set of three types. *Annals of Pure and Applied Mathematics*, 31(2):125–146, 2025.

[169] Bijan Davvaz and Thomas Vougiouklis. *Walk Through Weak Hyperstructures, A: Hv-structures*. World Scientific, 2018.

[170] K Hila and T Vougiouklis. Weak hyperstructures in lie-santilli admissible. *Journal of Algebraic Hyperstructures and Logical Algebras*, pages 1–13, 2025.

[171] Th Vougiouklis, S Spartalis, and M Kessoglides. Weak hyperstructures on small sets. *Ratio Mathematica*, 12(1):90–96, 1997.

[172] Thomas Vougiouklis. From structures to weak hyperstructures. *Journal of Mahani Mathematical Research*, 15(1):1–16, 2025.

[173] Souzana Vougioukli. Helix hyperoperation in teaching research. *Science & Philosophy*, 8(2):157–163, 2020.

[174] Souzana Vougioukli. Hyperoperations defined on sets of s-helix matrices. *Journal of Algebraic Hyperstructures and Logical Algebras*, 1(3), 2020.

[175] Piergiulio Corsini and Violeta Leoreanu. *Applications of hyperstructure theory*, volume 5. Springer Science & Business Media, 2013.

[176] Ajoy Kanti Das, Rajat Das, Suman Das, Bijoy Krishna Debnath, Carlos Granados, Bimal Shil, and Rakhal Das. A comprehensive study of neutrosophic superhyper bci-semigroups and their algebraic significance. *Transactions on Fuzzy Sets and Systems*, 8(2):80, 2025.

[177] Adel Al-Odhari. A brief comparative study on hyperstructure, super hyperstructure, and n-super superhyperstructure. *Neutrosophic Knowledge*, 6:38–49, 2025.

[178] Florentin Smarandache. *Hyperuncertain, superuncertain, and superhyperuncertain sets/logics/probabilities/statistics*. Infinite Study, 2017.

[179] Florentin Smarandache. Foundation of superhyperstructure & neutrosophic superhyperstructure. *Neutrosophic Sets and Systems*, 63(1):21, 2024.

[180] F. Smarandache. Introduction to superhyperalgebra and neutrosophic superhyperalgebra. *Journal of Algebraic Hyperstructures and Logical Algebras*, 2022.

[181] Michael Robinson. Abstract simplicial complexes. In *Computational Homological Algebra*, pages 199–246. Springer, 2026.

[182] Christopher T Lee, John B Moody, Rommie E Amaro, J Andrew McCammon, and Michael J Holst. The implementation of the colored abstract simplicial complex and its application to mesh generation. *ACM Transactions on Mathematical Software (TOMS)*, 45(3):1–20, 2019.

[183] Vladimir V Kornyak. Discrete relations on abstract simplicial complexes. *Programming and Computer Software*, 32(2):84–89, 2006.

[184] Gruia Clinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40:1740–1766, 2011.

[185] Ho Yee Cheung, Lap Chi Lau, and Kai Man Leung. Algebraic algorithms for linear matroid parity problems. *ACM Transactions on Algorithms (TALG)*, 10(3):1–26, 2014.

[186] Roy Goetschel and William Voxman. Fuzzy matroid sums and a greedy algorithm. *Fuzzy Sets and Systems*, 37:189–200, 1990.

[187] Shaher Momani, Viet-Thanh Pham, and Zhouchao Wei. Directed simplicial complexes in brain real-world networks. *The European Physical Journal Special Topics*, 233:807 – 816, 2024.

[188] Manuel Lecha, Andrea Cavallo, Francesca Dominici, Elvin Isufi, and Claudio Battiloro. Higher-order topological directionality and directed simplicial neural networks. *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2024.

[189] Manuel Lecha, Andrea Cavallo, Francesca Dominici, Ran Levi, Alessio Del Bue, Elvin Isufi, Pietro Morerio, and Claudio Battiloro. Directed semi-simplicial learning with applications to brain activity decoding. *ArXiv*, abs/2505.17939, 2025.

[190] Greg Friedman. Survey article: an elementary illustrated introduction to simplicial sets. *The Rocky Mountain Journal of Mathematics*, pages 353–423, 2012.

[191] Jon M Beck. Simplicial sets and the foundations of analysis. In *Applications of Sheaves: Proceedings of the Research Symposium on Applications of Sheaf Theory to Logic, Algebra, and Analysis, Durham, July 9–21, 1977*, pages 113–124. Springer, 2006.

[192] Samuel Peltier, Laurent Fuchs, and Pascal Lienhardt. Simploidals sets: Definitions, operations and comparison with simplicial sets. *Discrete Applied Mathematics*, 157(3):542–557, 2009.

[193] Mustafa Hajij, Kyle Istvan, and Ghada Zamzmi. Cell complex neural networks. *arXiv preprint arXiv:2010.00743*, 2020.

[194] Stefania Sardellitti and Sergio Barbarossa. Topological signal processing over generalized cell complexes. *IEEE Transactions on Signal Processing*, 72:687–700, 2024.

[195] John C Hart. Using the cw-complex to represent the topological structure of implicit surfaces and solids. In *ACM SIGGRAPH 2005 Courses*. Association for Computing Machinery, 2005.

[196] Mike Catanzaro. A user's guide: Dynamics and fluctuations of cellular cycles on cw complexes. *Enchiridion: Mathematical User's Guides*, 2, 2016.

[197] Martin R Bridson and André Haefliger. M κ-polyhedral complexes. In *Metric Spaces of Non-positive Curvature*, pages 97–130. Springer, 1999.

[198] Arturs Berzins. Polyhedral complex extraction from relu networks using edge subdivision. In *International conference on machine learning*, pages 2234–2244. PMLR, 2023.

[199] Prashant Gupta and Bala Krishnamoorthy. Euler transformation of polyhedral complexes. *International Journal of Computational Geometry & Applications*, 30(03n04):183–211, 2020.

[200] Dominic Desjardins Côté. Reduction of simplicial complex by relation and dowker complex. *arXiv preprint arXiv:2401.08475*, 2024.

[201] Marius Huber and Patrick Schnider. Flagifying the dowker complex. *arXiv preprint arXiv:2508.08025*, 2025.

[202] Jae Won Choi, Yuzhou Chen, José Frías, Joel Castillo, and Yulia Gel. Revisiting link prediction with the dowker complex. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 418–430. Springer, 2024.

[203] Gabor Hetyei. *Simplicial and cubical complexes: anologies and differences*. PhD thesis, Massachusetts Institute of Technology, 1994.

[204] Michel Deza, Mathieu Dutour, and Mikhail Shtogrin. On simplicial and cubical complexes with short links. *Israel journal of mathematics*, 144(1):109–124, 2004.

[205] Masahiro Hachimori. Orientations on simplicial complexes and cubical complexes. *Preprint, University of Tsukuba*, 2006.

[206] Quang Truong and Peter Chin. Weisfeiler and lehman go paths: Learning topological features via path complexes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 15382–15391, 2024.

[207] Seyed Mohammad Ajdani. Algebraic properties of the path complexes of cycles. *Mathematical Combinatorics*, 4:61–69, 2019.

[208] Xue Gong, Desmond J Higham, Konstantinos Zygalakis, and Ginestra Bianconi. Higher-order connection laplacians for directed simplicial complexes. *Journal of Physics: Complexity*, 5(1):015022, 2024.

[209] Steven Klee and Isabella Novik. From flag complexes to banner complexes. *SIAM Journal on Discrete Mathematics*, 27(2):1146–1158, 2013.

[210] Matthew Kahle. Sharp vanishing thresholds for cohomology of random flag complexes. *Annals of Mathematics*, pages 1085–1107, 2014.

[211] Daniel Lütgehetmann, Dejan Govc, Jason P Smith, and Ran Levi. Computing persistent homology of directed flag complexes. *Algorithms*, 13(1):19, 2020.

[212] Dejan Govc. Computing homotopy types of directed flag complexes. *arXiv preprint arXiv:2006.05333*, 2020.

[213] Florian Unger and Jonathan Krebs. Mcmc sampling of directed flag complexes with fixed undirected graphs. *Journal of Applied and Computational Topology*, 8(6):1881–1916, 2024.

[214] Jakob Hansen and Robert Ghrist. Toward a spectral theory of cellular sheaves. *Journal of Applied and Computational Topology*, 3(4):315–358, 2019.

[215] Chuan-Shen Hu. Cellular sheaves on higher-dimensional structures. *arXiv preprint arXiv:2505.23993*, 2025.

[216] Robert Ghrist and Hans Riess. Cellular sheaves of lattices and the tarski laplacian. *arXiv preprint arXiv:2007.04099*, 2020.

[217] Brendan J Frey, Frank R Kschischang, Hans-Andrea Loeliger, and Niclas Wiberg. Factor graphs and algorithms. In *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, volume 35, pages 666–680. Citeseer, 1997.

[218] Frank Dellaert. Factor graphs and gtsam: A hands-on introduction. *Georgia Institute of Technology, Tech. Rep*, 2(4), 2012.

[219] Uriel Feige and Shlomo Jozeph. Universal factor graphs. In *Automata, Languages, and Programming: 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I 39*, pages 339–350. Springer, 2012.

[220] Xiao-Yu Hu, Evangelos Eleftheriou, and D-M Arnold. Progressive edge-growth tanner graphs. In *GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No. 01CH37270)*, volume 2, pages 995–1001. IEEE, 2001.

[221] Xiao-Yu Hu, Evangelos Eleftheriou, and Dieter-Michael Arnold. Regular and irregular progressive edge-growth tanner graphs. *IEEE transactions on information theory*, 51(1):386–398, 2005.

[222] Tuvi Etzion, Ari Trachtenberg, and Alexander Vardy. Which codes have cycle-free tanner graphs? *IEEE Transactions on Information Theory*, 45(6):2173–2181, 2002.

[223] Manlio De Domenico. More is different in real-world multilayer networks. *Nature Physics*, 19(9):1247–1262, 2023.

[224] Manlio De Domenico, Albert Solé-Ribalta, Emanuele Cozzo, Mikko Kivelä, Yamir Moreno, Mason A Porter, Sergio Gómez, and Alex Arenas. Mathematical formulation of multilayer networks. *Physical Review X*, 3(4):041022, 2013.

[225] Mikko Kivelä, Alex Arenas, Marc Barthelemy, James P Gleeson, Yamir Moreno, and Mason A Porter. Multilayer networks. *Journal of complex networks*, 2(3):203–271, 2014.

[226] Petter Holme. Modern temporal network theory: a colloquium. *The European Physical Journal B*, 88(9):234, 2015.

[227] John Boaz Lee, Giang Nguyen, Ryan A Rossi, Nesreen K Ahmed, Eunyee Koh, and Sungchul Kim. Temporal network representation learning. *arXiv preprint arXiv:1904.06449*, 2019.

[228] Naoki Masuda and Petter Holme. Introduction to temporal network epidemiology. In *Temporal Network Epidemiology*, pages 1–16. Springer, 2017.

[229] Shengjie Liu, Jia Guo, and Chenyang Yang. Multidimensional graph neural networks for wireless communications. *IEEE Transactions on Wireless Communications*, 23(4):3057–3073, 2023.

[230] Mathieu Mach, Enrico Amico, Raphaël Liégeois, Maria Giulia Preti, Alessandra Griffa, Dimitri Van De Ville, and Mangor Pedersen. Connectome embedding in multidimensional graph spaces. *Network Neuroscience*, 8(4):1129–1148, 2024.

[231] Michael Rudolf, Hannes Voigt, Christof Bornhövd, and Wolfgang Lehner. Synopsys: Foundations for multidimensional graph analytics. In *International Workshop on Business Intelligence for the Real-Time Enterprise*, pages 159–166. Springer, 2013.

[232] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 793–803, 2019.

[233] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The world wide web conference*, pages 2022–2032, 2019.

[234] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, pages 2704–2710, 2020.

[235] Yongkang Li, Zipei Fan, Jixiao Zhang, Dengheng Shi, Tianqi Xu, Du Yin, Jinliang Deng, and Xuan Song. Heterogeneous hypergraph neural network for friend recommendation with human mobility. *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022.

[236] Xiangguo Sun, Hongzhi Yin, Bo Liu, Hongxu Chen, Jiuxin Cao, Yingxia Shao, and Nguyen Quoc Viet Hung. Heterogeneous hypergraph embedding for graph classification. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 725–733, 2021.

[237] Bilal Khan, Jia Wu, Jian Yang, and Xiaoxiao Ma. Heterogeneous hypergraph neural network for social recommendation using attention network. *ACM Transactions on Recommender Systems*, 3(3):1–22, 2025.

[238] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. Knowledge graphs. *ACM Computing Surveys (Csur)*, 54(4):1–37, 2021.

[239] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2015.

[240] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems*, 33(2):494–514, 2021.

[241] Bikram Pratim Bhuyan, Thipendra P Singh, Ravi Tomar, and Amar Ramdane-Cherif. Nesykhg: neuro-symbolic knowledge hypergraphs. *Procedia Computer Science*, 235:1278–1288, 2024.

[242] Zirui Chen, Xin Wang, Chenxu Wang, and Jianxin Li. Explainable link prediction in knowledge hypergraphs. In *Proceedings of the 31st ACM international conference on information & knowledge management*, pages 262–271, 2022.

[243] Mengfan Li, Xuanhua Shi, Chenqi Qiao, Teng Zhang, and Hai Jin. Hyperbolic hypergraph neural networks for multi-relational knowledge hypergraph representation. *arXiv preprint arXiv:2412.12158*, 2024.

[244] Bahare Fatemi, Perouz Taslakian, David Vazquez, and David Poole. Knowledge hypergraphs: Prediction beyond binary relations. *arXiv preprint arXiv:1906.00137*, 2019.

[245] Takaaki Fujita. Medical knowledge hypergraphs and medical knowledge superhypergraphs. *Annals of Healthcare Systems Engineering*, 877:878, 2025.

[246] Takaaki Fujita. Knowledge superhypergraphs, multimodal superhypergraphs, lattice superhypergraphs, and hyperbolic superhypergraphs: Concepts and applications. *Journal of Operational and Strategic Analytics*, 3(2):95–119, 2025.

[247] Zbigniew Suraj and Sibasis Bandyopadhyay. Generalized weighted fuzzy petri net in intuitionistic fuzzy environment. In *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 2385–2392. IEEE, 2016.

[248] Xuefei Lin, Xiao Chang, Yizheng Zhang, Zhanyu Gao, and Xu Chi. Automatic construction of petri net models for computational simulations of molecular interaction network. *NPJ Systems Biology and Applications*, 10(1):131, 2024.

[249] Hua Shi, Hu-Chen Liu, Jing-Hui Wang, and Xun Mou. New linguistic z-number petri nets for knowledge acquisition and representation under large group environment. *International Journal of Fuzzy Systems*, 24(8):3483–3500, 2022.

[250] Wei Jiang, Kai-Qing Zhou, Arezoo Sarkheyli-Hägele, and Azlan Mohd Zain. Modeling, reasoning, and application of fuzzy petri net model: a survey. *Artificial Intelligence Review*, 55(8):6567–6605, 2022.

[251] Ya-Xuan Yu, Hua-Ping Gong, Hu-Chen Liu, and Xun Mou. Knowledge representation and reasoning using fuzzy petri nets: a literature review and bibliometric analysis. *Artificial Intelligence Review*, 56(7):6241–6265, 2023.

[252] Jamal Khudair Madhloom, Zainab Hammoodi Noori, Sif K Ebis, Oday A Hassen, and Saad M Darwish. An information security engineering framework for modeling packet filtering firewall using neutrosophic petri nets. *Computers*, 12(10):202, 2023.

[253] Weichao Yue, Xiaoxue Wan, Sanyi Li, Hangli Ren, and Hui He. Simplified neutrosophic petri nets used for identification of superheat degree. *International Journal of Fuzzy Systems*, 24(8):3431–3455, 2022.

[254] Maribel Fernández, Hélène Kirchner, and Bruno Pinaud. Strategic port graph rewriting: an interactive modelling framework. *Mathematical Structures in Computer Science*, 29(5):615–662, 2019.

[255] Maribel Fernández, Hélène Kirchner, and Bruno Pinaud. Labelled port graph–a formal structure for models and computations. *Electronic Notes in Theoretical Computer Science*, 338:3–21, 2018.

[256] Maribel Fernández and János Varga. Finding candidate keys and 3nf via strategic port graph rewriting. In *Proceedings of the 22nd International Symposium on Principles and Practice of Declarative Programming*, pages 1–14, 2020.

[257] Maribel Fernández, Bruno Pinaud, and János Varga. A port graph rewriting approach to relational database modelling. In *International Symposium on Logic-Based Program Synthesis and Transformation*, pages 211–227. Springer, 2019.

[258] Dunwang Qin, Zhen Peng, and Lifeng Wu. Fchg: Fuzzy cognitive hypergraph for interpretable fault detection. *Expert Systems with Applications*, 255:124700, 2024.

[259] Wai Yu, Douglas Reid, and Stephen Brewster. Web-based multimodal graphs for visually impaired people. In *Universal Access and Assistive Technology: Proceedings of the Cambridge Workshop on UA and AT'02*, pages 97–108. Springer, 2002.

[260] Bruce N Walker, Michael A Nees, et al. An agenda for research and development of multimodal graphs. In *Proceedings of the International Conference on Auditory Display (ICAD2005)*, pages 428–432, 2005.

[261] Wai Yu and Stephen Brewster. Evaluation of multimodal graphs for blind people. *Universal access in the information society*, 2:105–124, 2003.

[262] Jian Huang, Kun Jiang, Yuanyuan Pu, Zhengpeng Zhao, Qiuxia Yang, Jinjing Gu, and Dan Xu. Multimodal hypergraph network with contrastive learning for sentiment analysis. *Neurocomputing*, page 129566, 2025.

[263] David López and Angélica Lozano. Shortest hyperpaths in a multimodal hypergraph with real-time information on some transit lines. *Transportation Research Part A: Policy and Practice*, 137:541–559, 2020.

[264] Fuhai Chen, Yue Gao, Donglin Cao, and Rongrong Ji. Multimodal hypergraph learning for microblog sentiment prediction. In *2015 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2015.

[265] Takaaki Fujita. Curried graph, curried hypergraph, and curried superhypergraph functions with applications to molecular structures, 2025.

[266] Grzegorz Bancerek. Curried and uncurried functions. *Formalized Mathematics*, 1(3):537–541, 1990.

[267] Takaaki Fujita and Florentin Smarandache. A unified framework for u-structures and functorial structure: Managing super, hyper, superhyper, tree, and forest uncertain over/under/off models. *Neutrosophic Sets and Systems*, 91:337–380, 2025.

[268] Takaaki Fujita and Florentin Smarandache. *A Dynamic Survey of Fuzzy, Intuitionistic Fuzzy, Neutrosophic, , and Extensional Sets*. Neutrosophic Science International Association (NSIA), 2025.

[269] Mohammed Alqahtani. Intuitionistic fuzzy quasi-supergraph integration for social network decision making. *International Journal of Analysis and Applications*, 23:137–137, 2025.

[270] T. Fujita, Muhammad Saqlain, and Muhammad Gulistan. Hierarchical network modeling with intuitionistic fuzzy superhypergraphs. *Multicriteria Algorithms with Applications*, 10(1):1–22, 2026.

[271] Shouxian Zhu. Neutrosophic n-superhypernetwork: A new approach for evaluating short video communication effectiveness in media convergence. *Neutrosophic Sets and Systems*, 85:1004–1017, 2025.

[272] E. J. Mogro, J. R. Molina, G. J. S. Canas, and P. H. Soria. Tree tobacco extract (*Nicotiana glauca*) as a plithogenic bioinsecticide alternative for controlling fruit fly (*Drosophila immigrans*) using n-superhypergraphs. *Neutrosophic Sets and Systems*, 74:57–65, 2024.

[273] Takaaki Fujita and Florentin Smarandache. Topological generalizations of graphs: Integrating hypergraph and superhypergraph perspectives. *Turkish Journal of Mathematics and Computer Science*, 17(2):322–337, 2025.

[274] Ovidiu Ivanciuc. Qsar comparative study of wiener descriptors for weighted molecular graphs. *Journal of chemical information and computer sciences*, 40(6):1412–1422, 2000.

[275] Sezer Sorgun and Kahraman Birgin. Vertex-edge-weighted molecular graphs: a study on topological indices and their relevance to physicochemical properties of drugs used in cancer treatment. *Journal of Chemical Information and Modeling*, 65(4):2093–2106, 2025.

[276] Zoiţa Berinde. Vertex-and edge-weighted molecular graphs for amines. *Revue Roumaine de Chimie*, 51(11):1131, 2006.

[277] Ahsanur Rahman, Christopher L Poirel, David J Badger, and TM Murali. Reverse engineering molecular hypergraphs. In *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, pages 68–75, 2012.

[278] Junwu Chen and Philippe Schwaller. Molecular hypergraph neural networks. *The Journal of Chemical Physics*, 160(14), 2024.

[279] Hiroshi Kajino. Molecular hypergraph grammar with its application to molecular optimization. *ArXiv*, abs/1809.02745, 2018.

[280] Hiroshi Kajino. Molecular hypergraph grammar with its application to molecular optimization. In *International Conference on Machine Learning*, pages 3183–3191. PMLR, 2019.

[281] Takaaki Fujita. Molecular fuzzy graphs, hypergraphs, and superhypergraphs. *Journal of Intelligent Decision and Computational Modelling*, 1(3):158–171, 2025.

[282] Takaaki Fujita. An introduction and reexamination of molecular hypergraph and molecular n-superhypergraph. *Asian Journal of Physical and Chemical Sciences*, 13(3):1–38, 2025.

[283] Zeeshan Saleem Mufti, Ali Tabraiz, Muhammad Farhan Hanif, et al. Molecular insights into tetracene through fuzzy topological indices in chemical graph theory. *Chemical Papers*, 79(5):2937–2953, 2025.

[284] Amer Marwan El-Samman, Incé Amina Husain, Mai Huynh, Stefano De Castro, Brooke Morton, and Stijn De Baerdemacker. Global geometry of chemical graph neural network representations in terms of chemical moieties. *Digital Discovery*, 3(3):544–557, 2024.

[285] Muhammad Aqib, Mehar Ali Malik, Hafiz Usman Afzal, Tahzeeb Fatima, and Yasir Ali. On topological indices of some chemical graphs. *Molecular Physics*, page e2276386, 2023.

[286] Daniel T Chang. Hypergraph: A unified and uniform definition with application to chemical hypergraph and more. *arXiv preprint arXiv:2405.12235*, 2024.

[287] Raffaella Mulas and Dong Zhang. Spectral theory of laplace operators on chemical hypergraphs. *arXiv preprint arXiv:2004.14671*, 2020.

[288] Elena Konstantinova. Chemical hypergraph theory. *Lecture Notes from Combinatorial & Computational Mathametics Center*, 2000.

[289] Takaaki Fujita. Chemical hypergraph and chemical superhypergraph. *Chemrxiv*, 2025.

[290] Takaaki Fujita. Note for biochemical/electrochemical/physicochemical hypergraphsand superhypergraphs. *Information Sciences and Technological Innovations*, 2025.

[291] Jiaxuan You, Bowen Liu, Rex Ying, Vijay S. Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. In *Neural Information Processing Systems*, 2018.

[292] Anand A. Rajasekar, Karthik Raman, and Balaraman Ravindran. Goal directed molecule generation using monte carlo tree search. *ArXiv*, abs/2010.16399, 2020.

[293] Tua A Tamba and Yul Y Nazaruddin. Data-driven construction of chemical reaction network graph using constrained lasso. In *2019 6th International Conference on Instrumentation, Control, and Automation (ICA)*, pages 226–230. IEEE, 2019.

[294] Hans G Othmer. A graph-theoretic analysis of chemical reaction networks. *Lecture Notes, Rutgers University*, 1:36, 1979.

[295] Gheorghe Craciun and Martin Feinberg. Multiple equilibria in complex chemical reaction networks: Ii. the species-reaction graph. *SIAM Journal on Applied Mathematics*, 66(4):1321–1338, 2006.

[296] Azriel Rosenfeld. Fuzzy graphs. In *Fuzzy sets and their applications to cognitive and decision processes*, pages 77–95. Elsevier, 1975.

[297] M. G. Karunambigai, R. Parvathi, and R. Buvaneswari. Arc in intuitionistic fuzzy graphs. *Notes on Intuitionistic Fuzzy Sets*, 17:37–47, 2011.

[298] N Vinoth Kumar and G Geetha Ramani. Product intuitionistic fuzzy graph. *International Journal of Computer Applications*, 28(1):31–33, 2011.

[299] Said Broumi, Mohamed Talea, Assia Bakali, and Florentin Smarandache. Single valued neutrosophic graphs. *Journal of New theory*, 10:86–101, 2016.

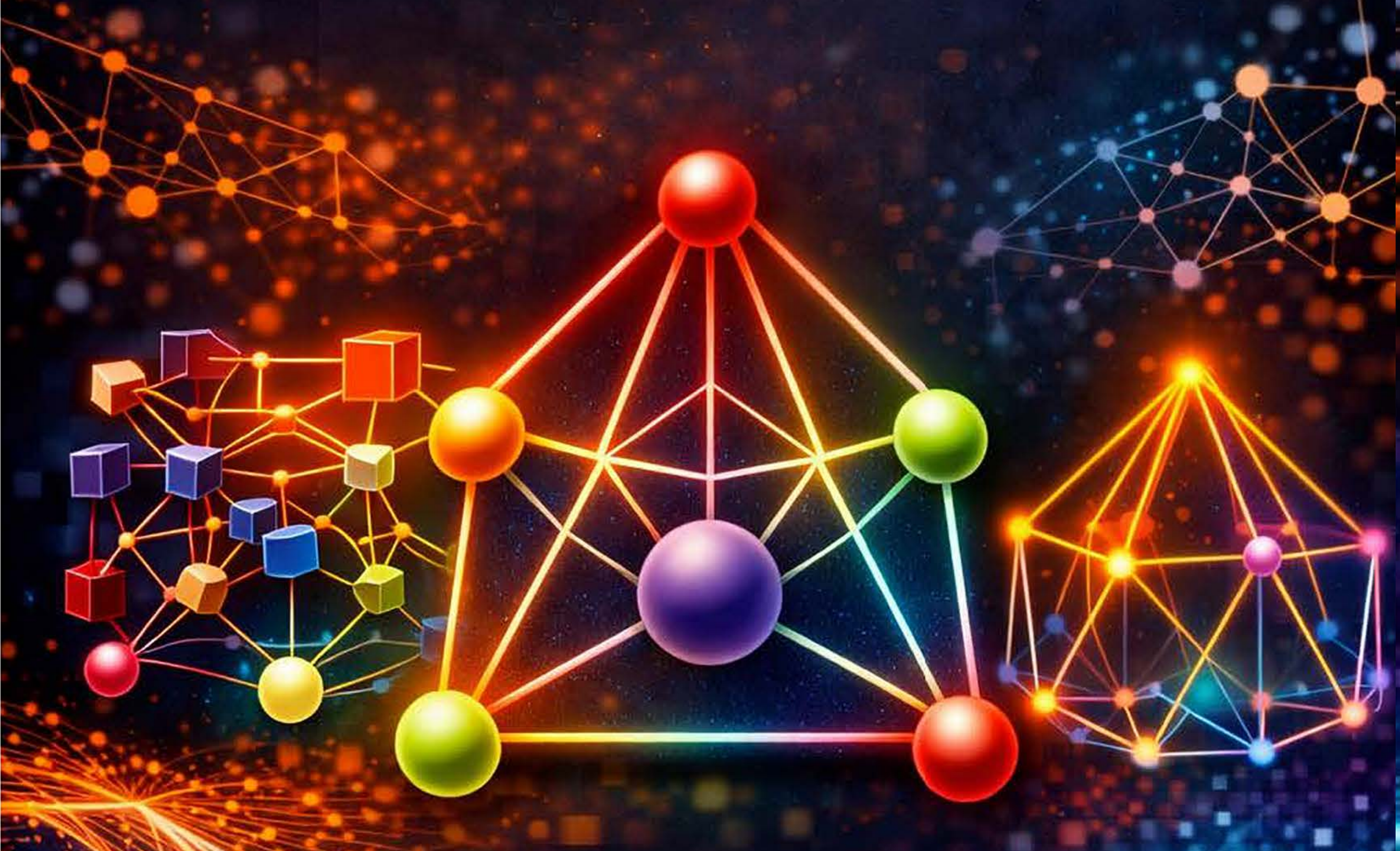
[300] Said Broumi, Assia Bakali, Mohamed Talea, Florentin Smarandache, and Prem Kumar Singh. Properties of interval-valued neutrosophic graphs. *Fuzzy Multi-criteria Decision-Making Using Neutrosophic Sets*, pages 173–202, 2019.

[301] Said Broumi, Mohamed Talea, Assia Bakali, and Florentin Smarandache. Interval valued neutrosophic graphs. *Critical Review, XII*, 2016:5–33, 2016.

[302] Prem Kumar Singh et al. Dark data analysis using intuitionistic plithogenic graphs. *International Journal of Neutrosophic Sciences*, 16(2):80–100, 2021.

[303] Fazeelat Sultana, Muhammad Gulistan, Mumtaz Ali, Naveed Yaqoob, Muhammad Khan, Tabasam Rashid, and Tauseef Ahmed. A study of plithogenic graphs: applications in spreading coronavirus disease (covid-19) globally. *Journal of ambient intelligence and humanized computing*, 14(10):13139–13159, 2023.

In this substantially expanded third edition, Fujta and Smarandache provide a comprehensive survey of graph-based frameworks for higher-order networks. The uptated edition not only corrects and improves earlier content but also introduces significantly new concepts and structures. It extends the scope with tensor-based models, recursive graph constructions, and advanced superhypergraph representations. This edition offers a unified theoretical framework that is deeper and more integrative, providing readers with powerful tools to navigate the complexities of modern networks.



Takaaki Fujita

Florentin Smarandache

9 781972 500220 0