Chapter 2
Mathematical Foundations of Hypergraph

Abstract In this chapter, we introduce the mathematical foundations of hypergraph and present the mathematical notations that are used to facilitate deep understanding and analysis of hypergraph structure. A hypergraph is composed of a set of vertices and hyperedges, and it is a generalization of a graph, where a weighted hypergraph quantifies the relative importance of hyperedges or vertices. Hypergraph can also be divided into two main categories, i.e., the undirected hypergraph representation and the directed hypergraph representation. The latter one further divides the vertices in one hyperedge into the source vertex set and the target vertex set to model more complex correlations. Additionally, we discuss the relationship between hypergraph and graph from the perspective of structural transformation and expressive ability. The most intuitive difference between a simple graph and a hypergraph can be observed in the size of order and expression of adjacency. A hypergraph can be converted into a simple graph using clique expansion, star expansion, and line expansion. Moreover, the proof based on random walks and Markov chains establishes the relationship between hypergraphs with edge-independent vertex weights and weighted graphs.

2.1 Introduction

The importance of high-order complex network modeling has been discussed in Chap. 1. In this chapter, we introduce the basic knowledge of hypergraph. In a hypergraph, the edge degree is usually higher than that of a simple graph, which is two for a simple graph. Different from a graph structure that can model pairwise connections with its 2-degree edges, a hypergraph can model correlations between practical data that are much more complex than pairwise relationships. As a result of its versatility and usefulness of modeling complex correlations of data, machine learning on hypergraph has attracted increasing attention.

Machine learning methods on hypergraph have been used in many real-world applications due to its advantages. A wide variety of tasks have been performed with hypergraph in computer vision, including image retrieval [1] and 3D object classification [2], video segmentation [3], re-identification of people [4], hyper-spectral

Q. Dai, Y. Gao, *Hypergraph Computation*, Artificial Intelligence: Foundations, Theory, and Algorithms, https://doi.org/10.1007/978-981-99-0185-2_2

image analysis [5], landmark retrieval [6], and visual tracking [7]. It is possible to embed a wide range of subjects into a hypergraph structure for these tasks. In different tasks, the hypergraph structure can be used to formulate the correlation among a variety of subjects. In image retrieval [3], the correlation among different images can be modeled in a hypergraph, where each vertex denotes an image and the hyperedges can be generated by finding similar image features. In 3D object classification [2], the correlation among different 3D objects can be modeled in a hypergraph, where each vertex denotes a 3D object and the hyperedges can be generated based on the similarity among these 3D objects. In person re-identification [4], a hypergraph structure can be constructed, where each vertex represents a personal image and the hyperedges can be generated based on the similarities in the feature space. Similar modeling attempts have been deployed in medical image analysis and bio-informatics studies to identify genes [8, 9], predict diseases [10, 11], identify sub-types [12], and analyze functional networks [13].

Before detailed introduction of the hypergraph computation paradigm, hypergraph modeling, and other related methods and applications, in this chapter, we first present preliminary knowledge of hypergraph and multiple representations of hypergraph. We also compare the hypergraph structure with the graph structure from four aspects.

2.2 Preliminary Knowledge of Hypergraph

The basic concepts of hypergraph are hereby briefly discussed. Table 2.1 provides the main notations and definitions of hypergraphs throughout this chapter. We first introduce undirected hypergraph and directed hypergraph, respectively, and then introduce the K-uniform hypergraph, probabilistic hypergraph, the relationship between hypergraph and bipartite graph, and the weights on hypergraph.

2.2.1 Undirected Hypergraph

Let $\mathscr{G}$ be an indication of a hypergraph (undirected hypergraph), which consists of a set of vertices $\mathscr{V}$ and a set of hyperedges $\mathscr{E}$. In a weighted hypergraph, each hyperedge $e \in \mathscr{E}$ is assigned with a weight $w(e)$, symbolizing the importance of the connection relationship throughout the whole hypergraph. Let $\mathbf{W}$ denote the diagonal matrix of the hyperedge weights, i.e., $\text{diag}(\mathbf{W}) = \left[w\left(e_1\right), w\left(e_2\right), \ldots, w\left(e_{|\mathscr{E}|}\right)\right]$. Given a hypergraph $\mathscr{G} = (\mathscr{V}, \mathscr{E}, \mathbf{W})$, the structure of the hypergraph is usually represented by an incidence matrix $\mathbf{H} \in \{0, 1\}^{|\mathscr{V}| \times |\mathscr{E}|}$, with each entry $\mathbf{H}(v, e)$ indicating whether the vertex v is in the hyperedge e:

$$\mathbf{H}(v, e) = \begin{cases} 1 & \text{if } v \in e \\ 0 & \text{if } v \notin e, \end{cases} \tag{2.1}$$

Table 2.1 Notations and definitions of hypergraphs

Notation	Definition
$\mathscr{G}$	The hypergraph
$\mathscr{V}$	The set of vertices
$\mathscr{E}$	The set of hyperedges
$\mathbf{W}$	The diagonal matrix of the hyperedge weights
$\mathbf{U}$	The diagonal matrix of the vertex weights
$\mathbf{X}$	The vertex feature matrix
$\mathbf{Y}$	The vertex label matrix
$\mathbf{H}$	The $\lvert\mathscr{V}\rvert \times \lvert\mathscr{E}\rvert$ incidence matrix of undirected hypergraph structure. $\mathbf{H}(v, e)$ indicates the connection strength between vertex v and hyperedge e
$\mathbf{D}_v$	The diagonal matrix of vertex degrees
$\mathbf{D}_e$	The diagonal matrix of hyperedge degrees
Δ	The Laplacian matrix of hypergraph
$\mathbf{x}_i$	The feature vector of vertex v_i
$d(v)$	The degree of vertex v
$\delta(e)$	The degree of hyperedge e
$w(e)$	The weight of hyperedge e
$u(v)$	The weight of vertex v

where $\mathbf{H}(v, e)$ indicates the possibility of vertex v assigned to hyperedge e or the importance of vertex v for hyperedge e. The degree of hyperedge e and the degree of vertex v are defined as follows:

$$\delta(e) = \sum_{v \in \mathscr{V}} \mathbf{H}(v, e), \tag{2.2}$$

and

$$d(v) = \sum_{e \in \mathscr{E}} w(e) * \mathbf{H}(v, e). \tag{2.3}$$

The traditional hypergraph structure creates associations among vertices, with a single hyperedge connecting multiple vertices that have associations. All vertices on the same hyperedge are given a value of 1 in the incidence matrix $\mathbf{H}$. The adjacency matrix $\mathbf{H}$ is calculated as in (2.1), whose elements are valued by 0 or 1. Each row represents each vertex in the hypergraph and the columns represent all hyperedges. Each column represents the set of vertices on this hyperedge.

Figure 2.1 shows an undirected hypergraph, including the hypergraph itself, the incidence matrix $\mathbf{H}$, the vertex set $\mathscr{V}$, the hyperedge set $\mathscr{E}$, and the weight matrix $\mathbf{W}$. In the illustrated undirected hypergraph, there are 3 hyperedges e_1, e_2, and e_3 with 6 vertexes. The degree of the hyperedge e_3 is 3, which contains vertices $\{v_3, v_4, v_6\}$. By the same token, other elements of $\mathbf{D}_v$ can be inferred. Vertex v_3 belongs to the hyperedges e_2 and e_3, and the degree of the vertex is 2. The incidence matrix $\mathbf{H}$ of

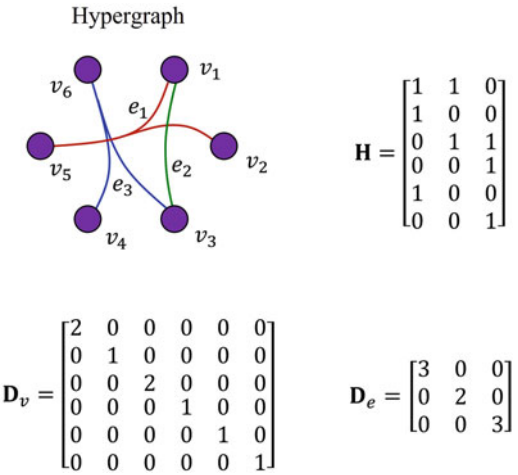


Fig. 2.1 An example of an undirected hypergraph

hypergraph is readily obtained by the rules of construction, which are shown on the right side of Fig. 2.1.

Given the incidence matrix $\mathbf{H}$ as calculated as in Eq. (2.1), all elements are valued by either 0 or 1. It is noted that the connection weights of different vertices on a hyperedge could be different. For example, some vertices are highly connected in the hyperedge and with high weights, while others may be with low weights. That is to say, the sum of each column of $\mathbf{H}$ is 1 (or not, due to different applications and objectives) and its values represent the vertex importance on this hyperedge.

There are various rules that can be used to determine whether vertices are associated with one another. Hyperedge groups can be generated from the data with a graph structure by using pairwise edges and k-hops; for the data without a graph structure, they can be generated by using neighbors in feature space. A detailed description of these methods is provided in Chap. 4.

2.2.2 Directed Hypergraph

The real world is incompatible with traditional undirected hypergraph representation in that hyperedges may be directional. Therefore, the representation of directed hypergraph structures is important. In each hyperedge, the vertex can be further divided into two sets: the source vertex set and the target vertex set. On directed hypergraph, a trivial definition [14] for the incidence matrix is defined as follows:

$$\hat{\mathbf{H}}(v, e) = \begin{cases} -1 & \text{if } v \in T(e) \\ 1 & \text{if } v \in S(e) \\ 0 & \text{otherwise,} \end{cases} \tag{2.4}$$

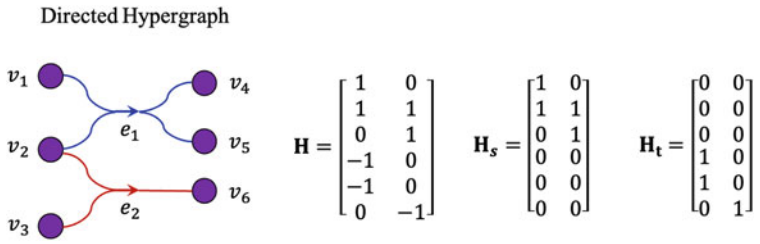


Fig. 2.2 An example of a directed hypergraph

where $T(e)$ and $S(e)$ are the target and source vertices for hyperedge e, respectively. The incidence matrix $\mathbf{H}$ is split into two matrices, $\mathbf{H_s}$ and $\mathbf{H_t}$, describing the source and target vertices for all hyperedges, respectively. When passing messages with these two incidence matrices, it is important to maintain the directional information. Two different incidence matrices guide message passing in the directed hypergraph, $\mathbf{H_s}$ and $\mathbf{H_t}$, unlike in the undirected hypergraph. The average aggregation of messages is normalized by $\mathbf{D_s}$ and $\mathbf{D_t}$ as two matrices, and it can be formulated as follows:

$$\begin{cases} \mathbf{D_s} = \text{diag}(\text{col\_sum}(\mathbf{H}_s)) \\ \mathbf{D_t} = \text{diag}(\text{col\_sum}(\mathbf{H}_t)), \end{cases} \tag{2.5}$$

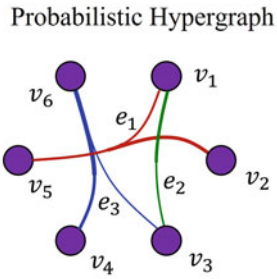
where $\text{diag}(v)$ is a function that converts a vector v to a diagonal matrix. The $\text{col\_sum}(\cdot)$ is a column accumulation function.

Figure 2.2 shows an example of directed hypergraph including the directed hypergraph itself, the incidence matrix $\mathbf{H}$, the source incidence matrix $\mathbf{H}_s$, and the target incidence matrix $\mathbf{H}_t$. The illustrated directed hypergraph contains six vertices and two hyperedge e_1 and e_2. e_1 connects four vertices and e_2 connects three vertices. In hyperedge e_1, the source vertices are v_1 and v_2, and the target vertices are v_4 and v_5. As for the hyperedge e_2, the source vertices are v_2 and v_3, and the target vertices are only v_6.

2.2.3 Probabilistic Hypergraph

In the real-world correlations, the intensity of the connection can not only be a binary number but also be a continuous number from zero to one. Consequently, the incidence matrix may be a continuous matrix with elements ranging from 0 to 1, which is adopted to denote a probabilistic hypergraph.

As shown in Fig. 2.3, the probabilistic hypergraph consists of six vertices and three hyperedges. The hyperedge e_1 connects three vertices v_1, v_2, and v_5. The intensity of the connection in this hyperedge is not the same. As shown in the right side of the figure, e_1 connects v_1 with an intensity of 0.3, connects v_2 with



$$\mathbf{H} = \begin{bmatrix} 0.3 & 0.7 & 0.0 \\ 0.8 & 0.0 & 0.0 \\ 0.0 & 0.3 & 0.2 \\ 0.0 & 0.0 & 0.8 \\ 0.5 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.9 \end{bmatrix}$$

$$\mathbf{D}_v = \begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.8 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.5 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.8 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.5 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.9 \end{bmatrix} \quad \mathbf{D}_e = \begin{bmatrix} 1.6 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.9 \end{bmatrix}$$

Fig. 2.3 An example of a probabilistic hypergraph

an intensity of 0.8, and connects v_5 with an intensity of 0.5. The degree of vertex and hyperedge in this type of hypergraph is computed by the sum of the row or column of the hypergraph incidence matrix **H**, as shown in the bottom of Fig. 2.3.

2.2.4 K-Uniform Hypergraph

In many applications, hyperedges in a hypergraph may connect the same number of vertices, which is known as the k-uniform hypergraph. In the k-uniform hypergraph, each hyperedge contains precisely k vertices, as shown in Fig. 2.4. Under this definition, a simple can be regarded as a spatial case of hypergraph, a 2-uniform hypergraph, where each hyperedge only connects two vertices.

Figure 2.4 illustrates an example of 3-uniform hypergraph. The hypergraph consists of six vertices and three hyperedges, and each hyperedge contains precisely 3 vertices. Hyperedge e_1 connects vertices v_1, v_2, and v_5. Hyperedge e_2 connects vertices v_1, v_2, and v_3. The degree of all hyperedges in this type of hypergraph is consistent k.

2.2.5 Hypergraph and Bipartite Graph

The bipartite graph can be indicated by $\mathscr{G} = \{\mathscr{U}, \mathscr{V}, \mathscr{E}\}$. Unlike the simple graph, vertices in the bipartite can be divided into two disjoint and independent sets $\mathscr{U}$

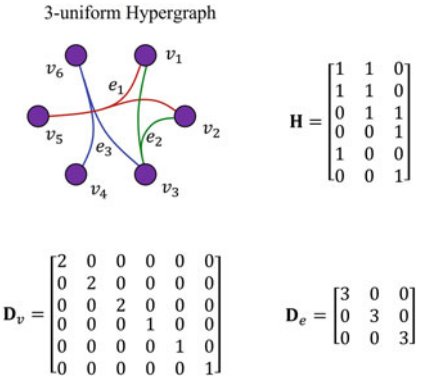


Fig. 2.4 An example of a 3-uniform hypergraph

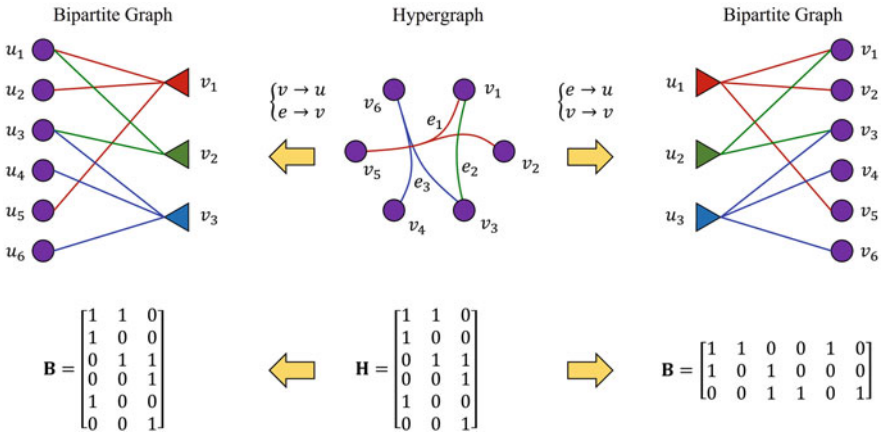


Fig. 2.5 The relationship between hypergraph and bipartite graph

and $\mathscr{V}$. Every edge only connects one vertex in set $\mathscr{U}$ and another vertex in set $\mathscr{V}$. Obviously, an undirected hypergraph can be regarded as a bipartite graph if the hyperedges are treated as another vertex set, as shown in Fig. 2.5.

Figure 2.5 illustrates examples of converting hypergraph to bipartite graph. The bipartite graph can be generated by two strategies: the vertices and hyperedges are treated as vertices in $\mathscr{U}$ and vertices in $\mathscr{V}$ (as illustrated in the left part), and the vertices and hyperedges are treated as vertices in $\mathscr{V}$ and vertices in $\mathscr{U}$ (as illustrated in the right part). Similarly, a bipartite graph can also be transformed to an undirected hypergraph with set $\mathscr{U}/\mathscr{V}$ as the hyperedges. It is not mean that the hypergraph is the same as or can be replaced with the bipartite graph. The transformation only exists in the undirected hypergraph and the probabilistic hypergraph. Confronting more complex hypergraph like directed hypergraph, the transformation will be invalid.

2.2.6 The Weights on Hypergraph

It is noted that there are different weights on a hypergraph, which provide additional information to assign values to a hypergraph structure. This is a more semantically preferred way of representing a hypergraph, as different components of a hypergraph, such as a vertex, a hyperedge or even a sub-hypergraph, should have different impact on the relationship modeling. For example, in a recommender system, the weights in the user profile influence the categorization of user attributes. If the attributes are not categorized accurately, the accuracy of the recommendations and marketing based on the profile could be questionable. The main types of weight information on a hypergraph are hyperedge weights and vertex weights, with the magnitude of the values indicating the relative importance of hyperedges and vertices, respectively.

First, let us show how the weights on vertex can be used. Different vertices may have varying importance on hypergraph modeling, and vertex weights are used in a hypergraph to determine the importance of different vertices. If a vertex is connected on the hypergraph strongly (with high correlations), it should be with a large vertex weight. Otherwise, it should be with a small vertex weight. For those vertices which have a 0 weight value in the incidence matrix, it can also be regarded as it is connected by the corresponding hyperedge with a weight of 0. Here, the diagonal elements of $\mathbf{U}$ to represent the weights of vertices, which are between 0 and 1, which reveal the relative importance of these hyperedges. Figure 2.6 shows an example hypergraph with vertex weights. In this figure, the weight of each vertex is denoted by the size of the vertex node. Vertex v_6 has a weight of 0.9, which is larger than all other vertices, and vertex v_2 is the smallest among the six vertices.

Then, let us focus on the weights on hyperedge. Hyperedge weights reflect the importance of different hyperedges in a hypergraph. As different hyperedges may have different importance in representing connections among vertices, it is crucial that hyperedges be weighted corresponding to their representative capabilities. In some cases, a part of hyperedges are more reliable due to its generation method or the features employed in this task, and these hyperedges should be given a large

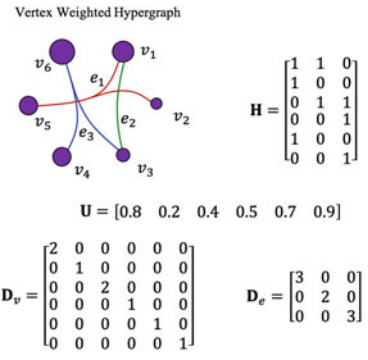


Fig. 2.6 An example of a vertex weighted hypergraph

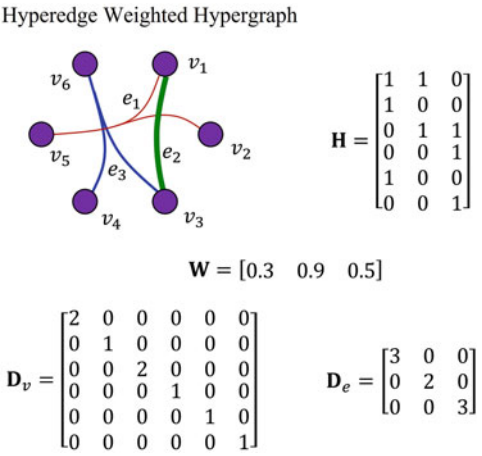


Fig. 2.7 An example of a hyperedge weighted hypergraph

weight during the learning process. Here, the diagonal element values of $\mathbf{W}$ can be used to represent the weights of vertices, which are between 0 and 1, revealing the relative importance of these hyperedges. Figure 2.7 shows an example of the hyperedge weighted hypergraph. In the illustration, the three hyperedges have the weights 0.3, 0.9, and 0.5, respectively.

2.3 Comparison Between Graph and Hypergraph

As a generalization of graph, the relationship between graph and hypergraph is a fundamental question. In this part, we detailedly introduce the relationship between graph and hypergraph from four aspects, i.e., the order of correlations, the representation methods, the structure transformation and random work on both of them.

2.3.1 Low-Order Versus High-Order Correlations

First, we define the *interaction* as a set $I = [p_0, p_1, \cdots, p_{k-1}]$ containing k basic elements of the system being studied, which can also be called vertices or nodes. Various real-world interactions can be described by such interactions, such as coauthors of a scientific paper, genes required to perform a specific function, neurons co-activating during a specific task, and more. We then denote the order (or dimension) of interactions among vertices as an order-0 interaction for a vertex interacting with itself only, an order-1 interaction for two vertices interacting with each other, an order-2 interaction for three vertices interactions, and so on.

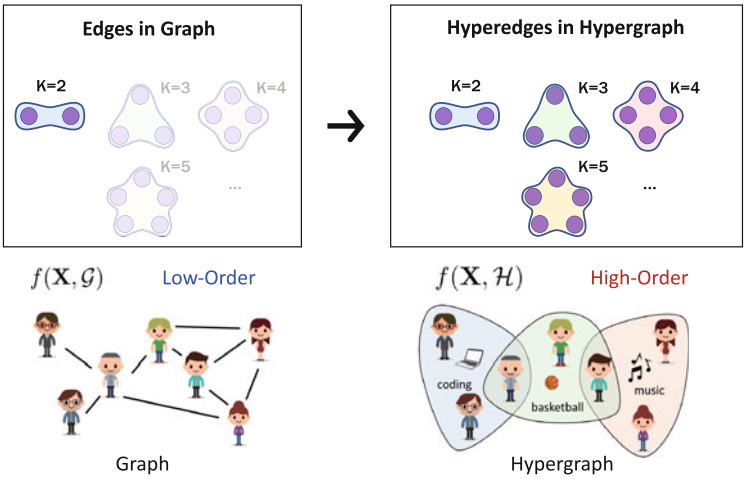


Fig. 2.8 The expressive ability comparison of graph and hypergraph

Furthermore, high-order interactions are considered k-interactions with $k \geq 2$. Low-order interactions, on the other hand, are those characterized by $k \leq 1$.

Figure 2.8 shows the comparison of hypergraph and graph on the modeling of different orders of correlations. We notice that a graph can only represent the order-1 interactions between two vertices. Different from graph, a hypergraph can represent any order-k interactions through its flexible hyperedges. From this direction, hypergraph is more effective on modeling high-order correlation among subjects compared with graph.

2.3.2 Adjacency Matrix Versus Incidence Matrix

A graph with N vertices can be described by an adjacency matrix $\mathbf{A} \in \{0, 1\}^{N \times N}$, where $\mathbf{A}_{i,j} = 1$ denotes that there is an edge connecting vertex v_i and vertex v_j. In most cases, the adjacency matrix $\mathbf{A}$ is a symmetry matrix.

A hypergraph with N vertices and M hyperedges can be described by an incidence matrix $\mathbf{H} \in \{0, 1\}^{N \times M}$, where $\mathbf{H}_{i,j} = 1$ denotes that the hyperedge e_j connects vertex v_i.

By comparison of adjacency matrix and incidence matrix, a graph can be regarded as a 2-uniform hypergraph. In this case, each hyperedge can only connect two vertices. Given the possible $N \times N$ order-1 hyperedges $\mathbf{H}$ in the 2-uniform hypergraph, they can be directly projected to the $N \times N$ elements in adjacency matrix $\mathbf{A}$. The hypergraph incidence and the simple graph adjacency matrix can be bi-transformed as follows:

$$\mathbf{H}\mathbf{H}^{\top} = \mathbf{A} + \mathbf{D}. \tag{2.6}$$

The adjacency matrix for graph and the incidence matrix for hypergraph have different processing styles when confronting multi-modal data or multiple types of connections. Given m adjacency matrices representing m graphs $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_m$, there are two typical ways to combine these data for graph. The first way is to combine different graphs into one graph $\mathcal{G}$ and then conduct other tasks. The second way is to conduct the task in each graph individually and then combine all these results. Figures 2.9 and 2.10 show these two types of methods. In either method, it is required to perform fusion, either in the graph structure part or in the result part. In recent years, a series of graph fusion methods [15, 16] have been introduced, while it is still a challenging task to optimally combine different graphs. On the other side, the multi-modal graph fusion is also with high computational complexity, which may limit the applications on multi-modal data.

Different from the processing method in graph, hypergraph can handle such types of different connections in an easy and direct way, due to its flexible hyperedges. As shown in Fig. 2.11, when there are multiple types of connections available, it is possible to generate multiple hyperedge groups with m incidence matrices $\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_m$, and these m incidence matrices can be directly concatenated to generate the overall hypergraph structure $\mathbf{H}$. In this way, all these multi-modal data or multiple types of connections can be easily modeled in one hypergraph and all further processing can be directly deployed on this hypergraph structure. Under such circumstances, it is not required to conduct multi-modal fusion in an explicit way, while it could be jointly included in the hypergraph computation process.

2.3.3 Structure Transformation from Hypergraph to Graph

A hypergraph can encode high-order data correlation (beyond pairwise) using its degree-free hyperedges compared to a simple graph, where the degree for all edges has to be 2. In a sense, a simple graph can be viewed as a special case, where all hyperedges on a hypergraph are of degree 2. Therefore, hypergraph and graph are interconvertible. Currently, there are a number of methods for converting a hypergraph to a simple graph. The common ones are clique expansion, star expansion, and line expansion, which are shown in Figs. 2.12, 2.13 and 2.14, respectively.

(1) Clique Expansion

Figure 2.12 shows an example of transforming a hypergraph to a graph with clique expansion. The clique expansion algorithm constructs a graph $\mathcal{G}^x(\mathcal{V}, E^x)$ from the original hypergraph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ by replacing each hyperedge e with edges, whose degree is 2, for each pair (u, v) of vertices in the hyperedge [17]: $\mathcal{E}^x = \{(u, v) : u, v \in e, e \in \mathcal{E}\}$.

It is interesting to note that the vertices in hyperedge e form a clique in the graph $\mathcal{G}^x$, exactly where the name comes from. $\mathcal{G}^x$ preserves the structure of the vertices of $\mathcal{G}$, so that the information on the edges needs to be reduced as far as possible to

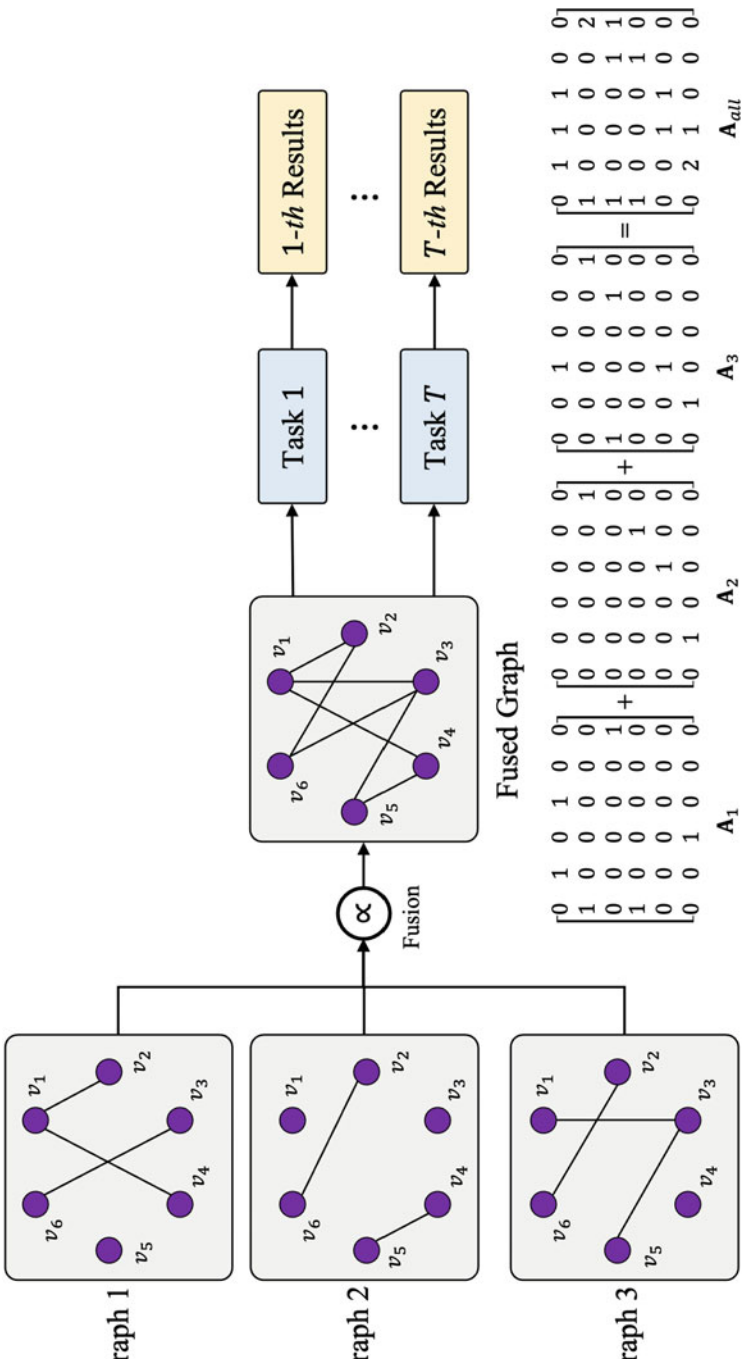


Fig. 2.9 An example of the graph structure fusion for the multi-modal data

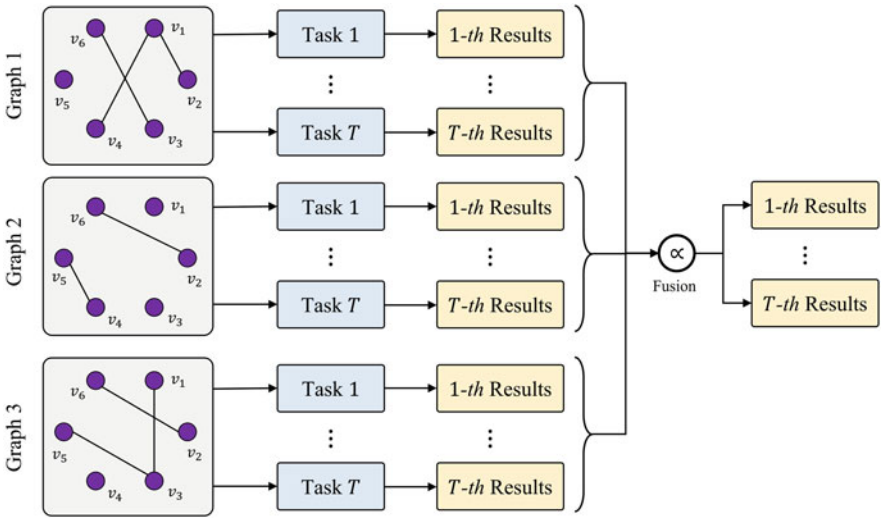


Fig. 2.10 An example of the results fusion for the multi-modal data

the higher order associations of the hyperedges. That is, the difference between the weights of any two edges that contains both u and v on $\mathscr{G}^x$ and the weights of the hyperedge connections should be as small as possible. Thus we use the following formula when assigning weights $w^x(u, v)$ to edges on $\mathscr{G}^x$:

$$w^x(u, v) = \underset{w^x(u,v)}{\arg\min} \sum_{e \in \mathscr{E}: u, v \in e} \left(w^x(u, v) - w(e)\right)^2. \tag{2.7}$$

Hence, clique expansion uses the discriminative model, where every edge in the clique of $\mathscr{G}^x$ associated with hyperedge e has weight $w(e)$. This criterion has the following minimizer:

$$w^x(u, v) = \mu \sum_{e \in \mathscr{E}: u, v \in e} w(e) = \mu \sum_{e} h(u, e) h(v, e) w(e), \tag{2.8}$$

where μ is a fixed scalar. Equivalently, from the point of view of edges, the weight between two vertices u and v is derived from the sum of the weights assigned by the hyperedge that contains all of them simultaneously.

(2) Star Expansion

Figure 2.13 shows an example of transforming a hypergraph to a graph with star expansion. By star expansion, a graph $\mathscr{G}^*(\mathscr{V}^*, \mathscr{E}^*)$ can be constructed from hypergraph $\mathscr{G}(\mathscr{V}, \mathscr{E})$ by regarding every hyperedge $e \in \mathscr{E}$ as a new vertex, thus $\mathscr{V}^* = \mathscr{V} \cup \mathscr{E}$ [17]. Each vertex in the hyperedge is connected to the new graph vertex e, i.e., $\mathscr{E}^* = \{(u, e) : u \in e, e \in \mathscr{E}\}$.

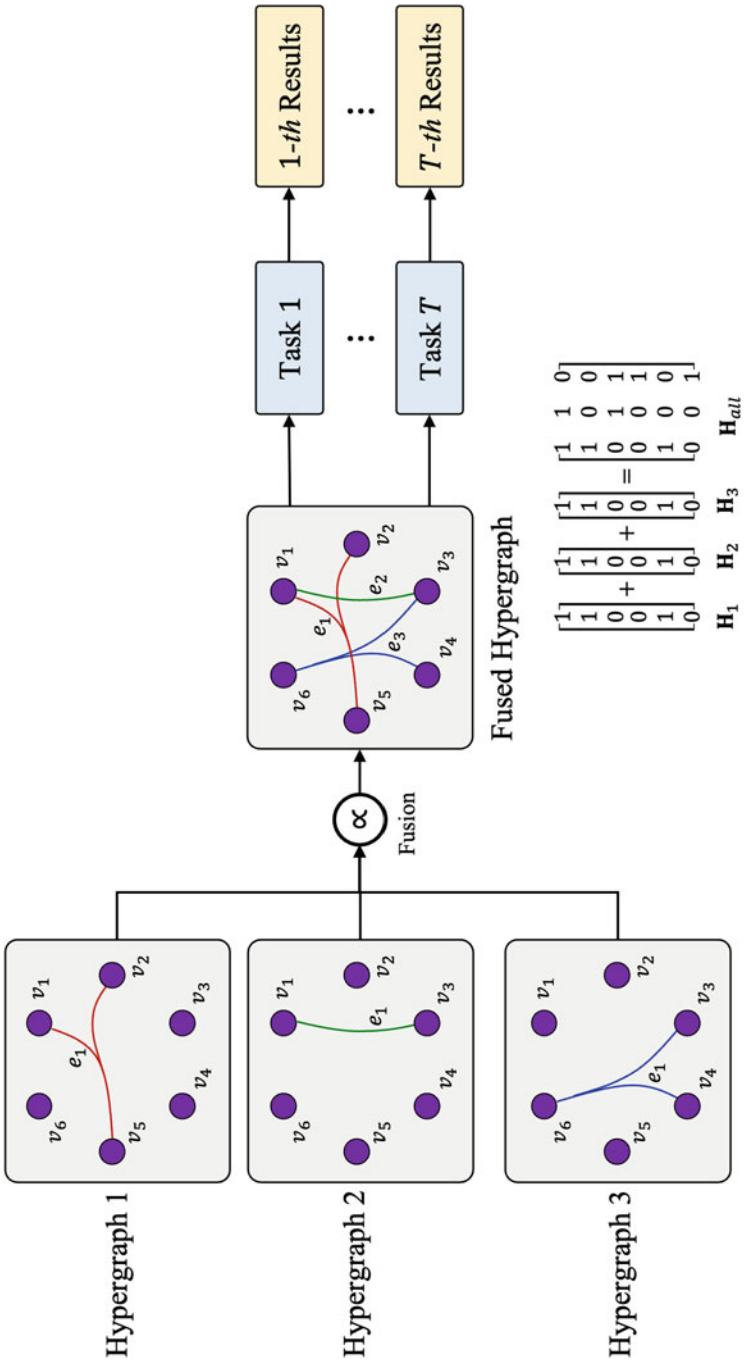


Fig. 2.11 An example of the hypergraph structure fusion for the multi-modal data

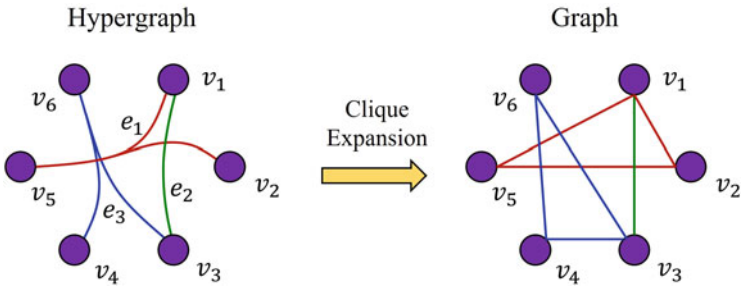


Fig. 2.12 An example of transforming a hypergraph to a graph with clique expansion

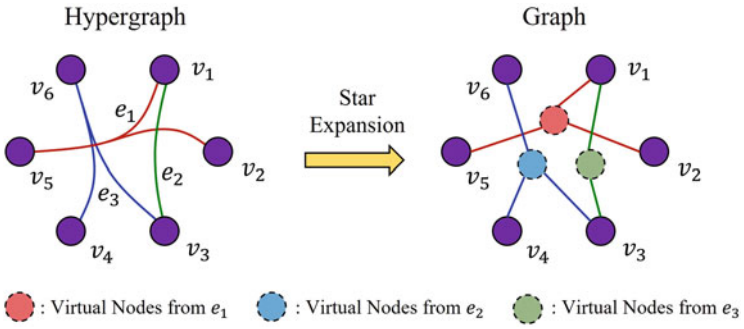


Fig. 2.13 An example of transforming a hypergraph to a graph with star expansion

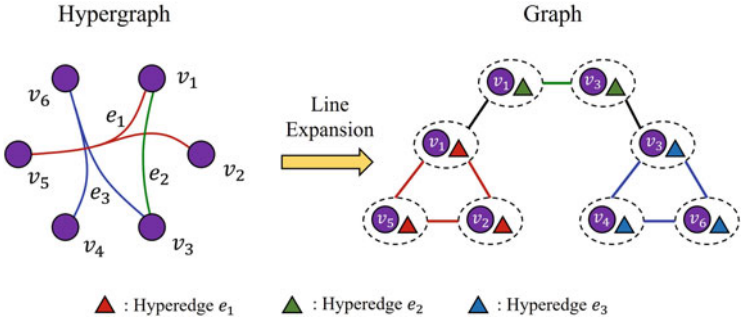


Fig. 2.14 An example of transforming a hypergraph to a graph with line expansion

There are different types of vertices in graph $\mathcal{G}^*$ and each hyperedge in $\mathcal{E}$ corresponds to a star in graph G. With star expansion, the scaled hyperedge weight is assigned to each graph edge $w^*(u, e)$ that corresponds to each hyperedge in $\mathcal{E}$ as follows:

$$w^*(u, e) = w(e)/\delta(e). \tag{2.9}$$

For each vertex representing a hyperedge, the weights of edges connecting to it are equivalent for equally dividing the superside weights into $|\delta(e)|$ parts.

(3) Line Expansion
Figure 2.14 shows an example of transforming a hypergraph to a graph with line expansion. In the case of line expansion algorithm, the vertices of the graph $\mathscr{G}^l = \left(\mathscr{V}^l, \mathscr{E}^l\right)$ are constructed by reconstructing the structure of the data stored in the vertices of the hypergraph, $\mathscr{G} = (\mathscr{V}, \mathscr{E})$. Each line vertex (u, e) in $\mathscr{G}^l$ can be viewed as a vertex in a context of a hyperedge or a hyperedge in a context of a vertex [18]. For each point on each hyperedge, a vertex is created to represent it. The vertex v in the line expended graph indicates the property of the vertex in the hyperedge, to each vertex in the hyperedge to it, i.e., $\mathscr{V}^* = \{(u, e) : u \in e, u \in \mathscr{V}, e \in \mathscr{E}\}$. This means that $\left|\mathscr{V}^l\right| = \sum_e \delta(e)$.

Therefore the vertexes in $\mathscr{G}^l$, which contain the same vertex or the same hyperedge, can be defined as the neighborhood. Consider both connections to be equally important, so $W^l = diag(1, \ldots, 1)$, $|W^l| = |\mathscr{V}^l| \times |\mathscr{V}^l|$. The mapping between a hypergraph $\mathscr{G}$ and its line expansion $\mathscr{G}^l$ is bijective under the construction.

2.3.4 Random Walks on Graph and Hypergraph

Random walks propagate the information stored in the vertices based on the links among the vertices in the graph or hypergraph. These links constitute the path of different vertices. In the hypergraph, each vertex's neighbor vertex messages are aggregated to update itself based on the "path" between the central vertex and each vertex in its neighborhood. A hypergraph's path between vertices v_1 and v_k is defined as a sequence, called hyperpath [19]:

$$P(v_1, v_k) = (v_1, e_1, v_2, e_2, \ldots, v_{k-1}, e_k, v_k), \tag{2.10}$$

where v_j and v_{j+1} are both part of the same vertex subset described by a hyperedge e_j. We say that a hyperpath separates two neighboring vertices by a hyperedge. In a hypergraph, messages between vertices are propagated through hyperedges, which are higher-order relationships than those in graphs. It is first necessary to extend the Neighbor Relation definition among vertices to the Inter-Neighbor Relation N over vertex set $\mathscr{V}$ and hyperedge set $\mathscr{E}$ for message propagation from vertex to hyperedge and hyperedge to hyperedge on the hyperpath.

Definition 1 The Inter-Neighbor Relation $N \subset \mathscr{V} \times \mathscr{E}$ on a hypergraph $\mathscr{G} = (\mathscr{V}, \mathscr{E}, \mathbf{W})$ with incidence matrix $\mathbf{H} \in \{0, 1\}^{|\mathscr{V}| \times |\mathscr{E}|}$ is defined as

$$N = \{\, (v, e) \mid \mathbf{H}(v, e) = 1,\ v \in \mathscr{V} \text{ and } e \in \mathscr{E} \,\}. \tag{2.11}$$

The hyperedge inter-neighbor set $N_e(v)$ of vertex v and the vertex inter-neighbor set $N_v(e)$ of hyperedge e are defined based on the Inter-Neighbor Relation.

Definition 2 The hyperedge inter-neighbor set of vertex $v \in \mathscr{V}$ is defined as

$$N_e(v) = \{ e \mid vNe,\ v \in \mathscr{V} \text{ and } e \in \mathscr{E} \}. \tag{2.12}$$

Definition 3 The vertex inter-neighbor set of hyperedge $e \in \mathscr{E}$ is defined as

$$N_v(e) = \{ v \mid vNe,\ v \in \mathscr{V} \text{ and } e \in \mathscr{E} \}. \tag{2.13}$$

With hypergraph learning, in contrast to graph learning, data are correlated at a higher level, and correlation models are expanded to a high level, resulting in improved performance in practice. This is just an apparent part of the nature of graph and hypergraph. Next, we delve deeper into the relationship between graphs and hypergraph from the point of view of mathematical derivations with the help of random walks [20] and Markov chain [21]. We then provide a mathematical comparison between hypergraph and graph. The proof concludes that, from random walks' aspect, a hypergraph with edge-independent vertex weights is equivalent to a weighted graph, and a hypergraph with edge-dependent vertex weights cannot be reduced to a weighted graph.

Two types of hypergraphs can be constructed to accurately represent real-world correlations, that is, hypergraph with vertex weights independent of edge and hypergraph with vertex weights dependent on edge. By using the binary hypergraph incidence matrix $\mathbf{H} \in \{0, 1\}^{|\mathscr{V}|\times|\mathscr{E}|}$, where vertices in each hyperedge share the same weight, hypergraph with edge-independent vertex weights ($\mathscr{G}_{\text{in}} = \{\mathscr{V}, \mathscr{E}, \mathbf{W}\}$) can model beyond pairwise correlations. Alternatively, the weighted hypergraph incidence matrix $\mathbf{R} \in \mathbb{R}^{|\mathscr{V}|\times|\mathscr{E}|}$ is used to model the variable correlation intensity in each hyperedge for the hypergraph with edge-dependent vertex weights ($\mathscr{G}_{\text{de}} = \{\mathscr{V}, \mathscr{E}, \mathbf{W}, \gamma\}$). We assume that hyperedge e includes vertex v, where $\gamma_e(v)$ denotes the connection intensity and $w(e)$ the weight of hyperedge e.

In hypergraph with edge-independent vertex weights, the definition of binary hypergraph incidence matrix $\mathbf{H}$, vertex degree $d(v)$, and hyperedge degree $\delta(e)$ is the same as in Sect. 2.1. In hypergraph with edge-dependent vertex weights, define the $d(v)$ and $\delta(e)$ as follows:

$$\begin{cases} d(v) = \sum\limits_{\beta \in \mathscr{N}_e(v)} w(\beta) \\ \delta(e) = \sum\limits_{\alpha \in \mathscr{N}_v(e)} \gamma_e(\alpha), \end{cases} \tag{2.14}$$

where $\mathscr{N}_v(\cdot)$ and $\mathscr{N}_e(\cdot)$ are defined in Eqs. (2.12) and (2.13), respectively.

Then, we will introduce the random walks and the Markov chain in hypergraph. First, we define the random walk in a hypergraph following papers [20–23]. At time t, a random walker at vertex v_t does the following:

- Pick an edge e containing vertex $v_t = v$, with probability $p_{v\to e}$.
- Pick vertex u from e, with probability $p_{e\to u}$.
- Move to vertex $v_{t+1} = u$, at time $t+1$.

We then define the transition probability $p_{v,u}$ of the corresponding Markov chain on $\mathcal{V}$ as $p_{v,u} = \sum_{e\in\mathcal{N}_e(v,u)} p_{v\to e}p_{e\to u}$, where $\mathcal{N}_e(v,u) = \mathcal{N}_e(v)\cap\mathcal{N}_e(u)$ denotes the hyperedge $\beta \in \mathcal{N}_e(v,u)$ containing vertices v and u, simultaneously. In hypergraph with edge-independent vertex weights, we have $p_{v\to e} = w(e)/d(v)$ and $p_{e\to u} = 1/\delta(e)$. The transition probability $p_{v,u}$ can be written as $p_{v,u} = \sum_{\beta\in\mathcal{N}_e(v,u)} \frac{w(\beta)}{d(v)}\cdot\frac{1}{\delta(\beta)}$. In hypergraph with edge-dependent vertex weights, we have $p_{v\to e} = w(e)/d(v)$ and $p_{e\to u} = \gamma_e(u)/\delta(e)$, and the transition probability $p_{v,u}$ can be written as $p_{v,u} = \sum_{\beta\in\mathcal{N}_e(v,u)} \frac{w(\beta)}{d(v)}\cdot\frac{\gamma_\beta(u)}{\delta(\beta)}$.

The following lemmas and definitions are used to compare the graph and the two types of hypergraphs [21].

Definition 4 Let M be a Markov chain with state space X and transition probability $p_{x,y}$, for $x, y \in S$. It can be said that M is reversible if there exists a probability distribution π over S such that $\pi_x p_{x,y} = \pi_y p_{y,x}$.

Lemma 5 *Let M be an irreducible Markov chain with finite state space S and transition probability $p_{x,y}$ for $x, y \in S$. M is reversible if and only if there exists a weighted undirected graph $\mathcal{G}$ with vertex set S such that random walks on $\mathcal{G}$ and M are equivalent.*

Proof of Lemma 5 Note that π indicates the stationary distribution [21, 24] of a given edge-independent/edge-dependent hypergraph. The transition probability $p_{v,u}$ of vertices in hypergraph with edge-independent vertex weights is defined as

$$p_{v,u} = \sum_{\beta\in\mathcal{N}_e(v,u)} \left(\frac{w(\beta)}{d(v)}\right)\left(\frac{1}{\delta(\beta)}\right). \tag{2.15}$$

Moreover, the transition probability $p_{v,u}$ of vertices in hypergraph with edge-dependent vertex weights is defined as

$$p_{v,u} = \sum_{\beta\in\mathcal{N}_e(v,u)} \left(\frac{w(\beta)}{d(v)}\right)\left(\frac{\gamma_\beta(u)}{\delta(\beta)}\right). \tag{2.16}$$

"⇒": Suppose M is reversible with transition probability $p_{x,y}$. We then construct a graph $\mathcal{G}$ with vertex set S and edge weights $w_{x,y} = \pi_x p_{x,y}$. Because M is irreducible, $\pi_x \neq 0$ and $p_{x,y} \neq 0$ for all states x and y. Thus, the edge weight $w_{x,y} \neq 0$ and the graph $\mathcal{G}$ are a connected graph. Due to the reversibility of M that $w_{x,y} = \pi_x p_{x,y} = \pi_y p_{y,x} = w_{y,x}$, the constructed graph $\mathcal{G}$ is an undirected graph. Random walks on $\mathcal{G}$ from x to y in one-time step satisfy the following:

$$\frac{w_{x,y}}{\sum_{z\in S} w_{x,z}} = \frac{\pi_x p_{x,y}}{\sum_{z\in S}\pi_x p_{x,z}} = \frac{p_{x,y}}{\sum_{z\in S} p_{x,z}} = p_{x,y}, \tag{2.17}$$

since $\sum_{z\in S} p_{x,z} = 1$. Thus, if M is reversible, the stated claim holds.

"⇐": Random walks on an undirected graph are always reversible.

Definition 6 A Markov chain is reversible if and only if its transition probability satisfies

$$p_{v_1,v_2} p_{v_2,v_3} \cdots p_{v_n,v_1} = p_{v_1,v_n} p_{v_n,v_{n-1}} \cdots p_{v_2,v_1} \tag{2.18}$$

for any finite sequence of states $v_1, v_2, \cdots v_n \in S$. The definition is also known as Kolmogorov's criterion. For more detailed proofs, please refer to [25].

Theorem 1 *Let $\mathscr{G}_{in} = \{\mathscr{V}, \mathscr{E}, \mathbf{W}\}$ be a hypergraph with edge-independent weights, and then there exists a weighted undirected graph $\mathscr{G}$ such that a random walk on $\mathscr{G}$ is equivalent to a random walk on $\mathscr{G}_{in}$.*

Proof of Theorem 1 The probability $p_{v,u}$ of $\mathscr{G}_{\text{in}}$ is defined in Eq. (2.15). By Definition 6, the following equation can be deduced:

$$\begin{aligned} & p_{v_1,v_2} p_{v_2,v_3} \cdots p_{v_n,v_1} \\ &= \sum_{\beta\in\mathscr{N}_e(v_1,v_2)} \left(\frac{w(\beta)}{d(v_1)} \cdot \frac{1}{\delta(\beta)}\right) \cdots \sum_{\beta\in\mathscr{N}_e(v_n,v_1)} \left(\frac{w(\beta)}{d(v_n)} \cdot \frac{1}{\delta(\beta)}\right) \\ &= \left(\frac{1}{d(v_1)} \sum_{\beta\in\mathscr{N}_e(v_1,v_2)} \frac{w(\beta)}{\delta(\beta)}\right) \cdots \left(\frac{1}{d(v_n)} \sum_{\beta\in\mathscr{N}_e(v_n,v_1)} \frac{w(\beta)}{\delta(\beta)}\right) \\ &= \frac{1}{d(v_2)} \sum_{\beta\in\mathscr{N}_e(v_1,v_2)} \frac{w(\beta)}{\delta(\beta)} \cdots \frac{1}{d(v_1)} \sum_{\beta\in\mathscr{N}_e(v_n,v_1)} \frac{w(\beta)}{\delta(\beta)}. \end{aligned} \tag{2.19}$$

For any v_i and v_j, $\sum_{\beta\in\mathscr{N}_e(v_i,v_j)} \frac{w(\beta)}{\delta(\beta)} = \sum_{\beta\in\mathscr{N}_e(v_j,v_i)} \frac{w(\beta)}{\delta(\beta)}$. Thus, the reversibility can be proven by

$$\begin{aligned} & p_{v_1,v_2} p_{v_2,v_3} \cdots p_{v_n,v_1} \\ &= \frac{1}{d(v_2)} \sum_{\beta\in\mathscr{N}_e(v_2,v_1)} \frac{w(\beta)}{\delta(\beta)} \cdots \frac{1}{d(v_1)} \sum_{\beta\in\mathscr{N}_e(v_1,v_n)} \frac{w(\beta)}{\delta(\beta)} \\ &= p_{v_2,v_1} p_{v_3,v_2} \cdots p_{v_1,v_n} \\ &= p_{v_1,v_n} p_{v_n,v_{n-1}} \cdots p_{v_2,v_1}. \end{aligned} \tag{2.20}$$

We say that a random walk on $\mathscr{G}_{\text{in}}$ is reversible. Furthermore, by Lemma 5, a random walk on $\mathscr{G}_{\text{in}}$ is equivalent to a random walk on a weighted undirected graph $\mathscr{G}$.

The proof of Theorem 1 can be processed as follows:

1. A random walk on $\mathscr{G}_{\text{in}}$ is equivalent to a random walk on a reversible Markov chain (according to **Definition 6**).

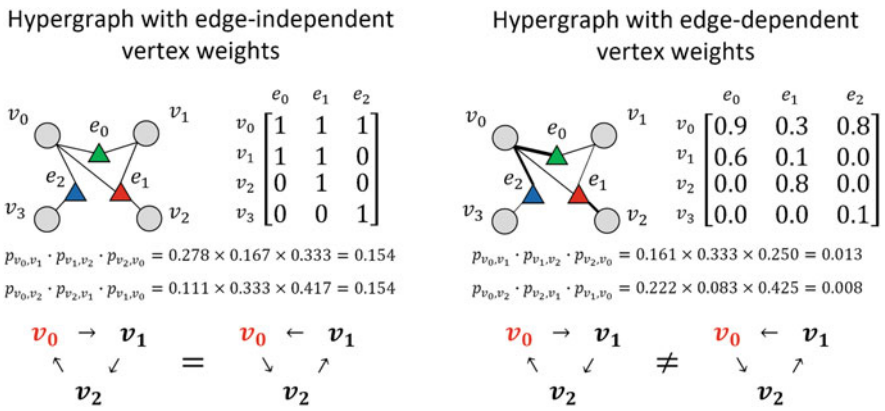


Fig. 2.15 An example of two types of random walks on the hypergraph with edge-independent vertex weights and the hypergraph with edge-dependent vertex weights. This figure is from [26]

2. A random walk on a reversible Markov chain is equivalent to a random walk on a weighted undirected graph $\mathcal{G}$ (according to **Lemma 5**).

Theorem 2 *Let $\mathcal{G}_{de} = \{\mathcal{V}, \mathcal{E}, \mathbf{W}, \gamma\}$ be a hypergraph with edge-dependent weights, and then there does not exist a weighted undirected graph $\mathcal{G}$ such that a random walk on $\mathcal{G}$ is equivalent to a random walk on $\mathcal{G}_{de}$.*

Proof of Theorem 2 Figure 2.15 provides an example that a random walk on $\mathcal{G}_{\text{de}}$ is not equivalent to a random walk on a reversible Markov chain. According to the second step of **Theorem 1**'s proof, **Theorem 2** holds.

A simple illustration is shown in Fig. 2.15 to make it easier to understand. There is no difference in the connection structure between the two hypergraphs, but there is a difference in the intensity of the connections. For two types of hypergraphs, the transition probabilities $p_{v,u}$ can be computed accordingly. As a consequence, two random walks from vertex v_0 are conducted: "$v_0 \to v_1 \to v_2 \to v_0$" and "$v_0 \to v_2 \to v_1 \to v_0$." Having obtained $p_{v_0,v_1} \cdot p_{v_1,v_2} \cdot p_{v_2,v_0}$ and $p_{v_0,v_2} \cdot p_{v_2,v_1} \cdot p_{v_1,v_0}$ for the two paths, the cumulative transition probability can then be calculated. This type of hypergraph is reversible according to **Theorem 1** and **Lemma 5**. Thus, from the two reversible paths, the same accumulated transition probability can be obtained. Alternatively, two different accumulated transition probabilities are obtained from two reversible paths in the hypergraph with edge-independent vertex weights.

2.4 Summary

In this chapter, we present the mathematical definition of the foundations of hypergraph and their interpretation. We then also show the representation of directed

hypergraph, different from undirected hypergraph, which represents the relationships between vertices within a hyperedge. Finally, we discuss the relationship between graph and hypergraph in conversions and expressive ability perspectives. The most intuitive differences between graph and hypergraph can be seen in low-order versus high-order representations and adjacency matrix versus incidence matrix. Clique expansion, star expansion, and line expansion are methods for converting hypergraph into simple graph. We also show the relationship between graph and hypergraph from the random walk view. A hypergraph with edge-independent vertex weights is equivalent to a weighted graph, and a hypergraph with edge-dependent vertex weights cannot be reduced to a weighted graph from the information propagation process on graph/hypergraph.

References

1. Y. Huang, Q. Liu, S. Zhang, D.N. Metaxas, Image retrieval via probabilistic hypergraph ranking, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2010), pp. 3376–3383
2. Y. Gao, M. Wang, D. Tao, R. Ji, Q. Dai, 3-D object retrieval and recognition with hypergraph analysis. IEEE Trans. Image Process. **21**(9), 4290–4303 (2012)
3. Y. Huang, Q. Liu, D. Metaxas, Video object segmentation by hypergraph cut, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2009), pp. 1738–1745
4. W. Zhao, S. Tan, Z. Guan, B. Zhang, M. Gong, Z. Cao, Q. Wang, Learning to map social network users by unified manifold alignment on hypergraph. IEEE Trans. Neural Netw. Learn. Syst. **29**(12) 5834–5846 (2018)
5. F. Luo, B. Du, L. Zhang, L. Zhang, D. Tao, Feature learning using spatial-spectral hypergraph discriminant analysis for hyperspectral image. IEEE Trans. Cyber. **49**(7), 2406–2419 (2018)
6. L. Zhu, J. Shen, H. Jin, R. Zheng, L. Xie, Content-based visual landmark search via multimodal hypergraph learning. IEEE Trans. Cyber. **45**(12), 2756–2769 (2015)
7. D. Du, H. Qi, L. Wen, Q. Tian, Q. Huang, S. Lyu, Geometric hypergraph learning for visual tracking. IEEE Trans. Cyber. **47**(12), 4182–4195 (2017)
8. Z. Tian, T. Hwang, R. Kuang, A hypergraph-based learning algorithm for classifying gene expression and arrayCGH data with prior knowledge. Bioinformatics. **25**(21), 2831–2838 (2009)
9. X. Zheng, W. Zhu, C. Tang, M. Wang, Gene selection for microarray data classification via adaptive hypergraph embedded dictionary learning. Gene. **706**, 188–200 (2019)
10. Y. Gao, C.-Y. Wee, M. Kim, P. Giannakopoulos, M.L. Montandon, S. Haller, D. Shen. MCI identification by joint learning on multiple MRI data, in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention* (2015), pp. 78–85
11. W. Shao, Y. Peng, C. Zu, M. Wang, D. Zhang, Hypergraph based multi-task feature selection for multimodal classification of Alzheimer's disease. Comput. Med. Imaging Graph. **80**, 101663 (2020)
12. E. Ramadan, S. Perincheri, D. Tuck, A hyper-graph approach for analyzing transcriptional networks in breast cancer, in *Proceedings of the ACM International Conference on Bioinformatics and Computational Biology* (2010), pp. 556–562
13. L. Xiao, J. Wang, P.H. Kassani, Y. Zhang, Y. Bai, J.M. Stephen, T.W. Wilson, V.D. Calhoun, Y. Wang, Multi-hypergraph learning based brain functional connectivity analysis in fMRI data. IEEE Trans. Med. Imaging **39**(5), 1746–1758 (2019)

14. G. Gallo, G. Longo, S. Pallottino, S. Nguyen, Directed hypergraphs and applications. Discrete Appl. Math. **42**(2–3), 177–201 (1993)
15. K. Zhan, C. Niu, C. Chen, F. Nie, C. Zhang, Y. Yang, Graph structure fusion for multiview clustering. IEEE Trans. Knowl. Data Eng. **31**(10), 1984–1993 (2018)
16. Z. Kang, G. Shi, S. Huang, W. Chen, X. Pu, J.T. Zhou, Z. Xu, Multi-graph fusion for multi-view spectral clustering. Knowl.-Based Syst. **189**, 105102 (2020)
17. Y.J. Zien, M. Schlag, P. Chan, Multilevel spectral hypergraph partitioning with arbitrary vertex sizes. IEEE Trans. Comput.-Aided Design Integr. Circ. Syst. **18**(9), 1389–1399 (1999)
18. C. Yang, R. Wang, S. Yao, T. Abdelzaher, Hypergraph learning with line expansion (2020). Preprint arXiv:2005.04843
19. R. Dharmarajan, K. Kannan, Hyper paths and hyper cycles. Ital. J. Pure Appl. Math. **98**(3), 309–312 (2015)
20. T. Carletti, F. Battiston, G. Cencetti, D. Fanelli, Random walks on hypergraphs, Phys. Rev. E **101**(2), 022308 (2020)
21. U. Chitra, B. Raphael, Random walks on hypergraphs with edge-dependent vertex weights, in *Proceedings of the Machine Learning Research* (2019), pp. 1172–1181
22. D. Zhou, J. Huang, B. Schölkopf, Learning with hypergraphs: Clustering, classification, and embedding, in *Proceedings of the Advances in Neural Information Processing Systems* (2007)
23. A. Ducournau, A. Bretto, Random walks in directed hypergraphs and application to semi-supervised image segmentation. Comput. Vision Image Understand. **120**, 91–102 (2014)
24. J. Li, J. He, Y. Zhu, E-tail product return prediction via hypergraph-based local graph cut, in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2018), pp. 519–527
25. P.F. Kelly, *Reversibility and Stochastic Networks* (Cambridge University Press, Cambridge, 2011)
26. Y. Gao, Y. Feng, S. Ji, R. Ji, HGNN+: General hypergraph neural networks. IEEE Trans. Pattern Analy. Mach. Intell. **45**(3), 3181–3199 (2023)