HYPERGRAPHS AND DATABASE IMPLEMENTATION

WRITTEN BY KENDYL DOUGLAS

Systems Imagination

BACKGROUND

In graph theory, a graph, *G(V,E)*, is a set of vertices, *V*, and an edge set, *E*, connects those vertices. This means that in a graph the edge set is a 2-element subset of V. For example, $G(V_t, E_t)$ where the vertex set V_t = {1,2,3} and the edge set E_t = {12, 23, 13}, is a triangle. In a graph, edges connect two vertices. In a hypergraph, edges can connect many vertices. Therefore, hypergraphs are generalizations of graphs where the edges are sets of two or more vertices

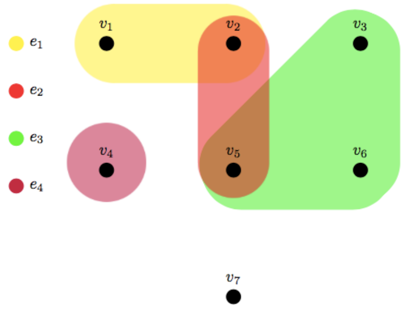


Figure 1

In Figure 1, there are 4 hyperedges which enclose regions (yellow, orange, green and purple). A given element (node) or entire hyperedge can be contained within another hyperedge and any given node (or hyperedge) can be a member of more than one hyper edge. Finally, any given node (or hyperedge) does not have to be a member of any other hyperedge.

Hypergraphs can be interpreted from a graph theoretical framework or as a family of sets (from *X*, the universal set). Both interpretations can provide value depending on the context. From a set-theoretical perspective, consider Sperner Families; which are families of sets in which no one set contains another. Sperner hypergraphs, or clutters, are hypergraphs where no hyperedge is a subset of another. , clutters are precisely Sperner families. From a graph theoretical perspective, hypergraph problems can be translated into a graph problem. Graphs have been much more explored than hypergraphs, so translating a hypergraph problem into a graph problem often makes a problem more readily solvable. For example, it can be proven that any non-trivial graph is the line-graph of a linear hypergraph. The proof is shown in [1] and the visual is shown in Figure 2. This allows graph theorists to interpret hypergraphs from a variety of perspectives within the largely explored field of graph theory.

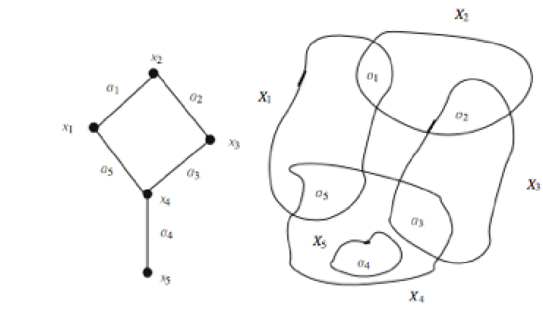


Fig. 2.2 Figure above illustrates Proposition 2.1

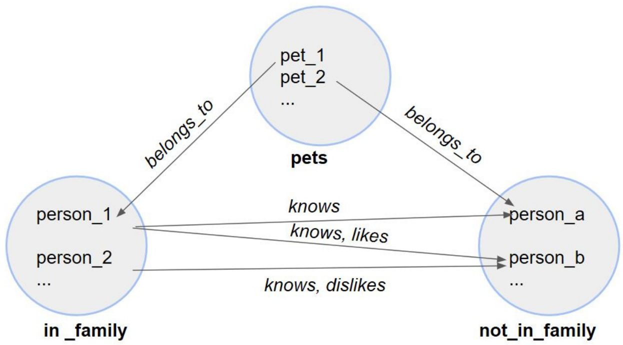
Illustration of how a non-trivial graph is the line-graph (right) of a linear hypergraph (left) [1].

Figure 2

Systems Imagination believes a hypergraph database can best characterize the data we receive from customers. The healthcare and life science data that SII works with has multiple levels of information, and a variety of perspectives are needed to grasp these complexities. We believe these data complexities can be most accurately modeled with a hypergraph database, yet there isn't really a straightforward way to implement a hypergraph database in SQL.

For example, directed hypergraph and undirected hypergraphs have been implemented very differently in other databases. (Directed hypergraphs are hypergraphs in which each edge has a direction. In this type of hypergraph, an edge may go from vertex *i* to vertex *j*, but not from vertex *j* to vertex *i*. Unlike in a directed hypergraph, an undirected hypergraph is a hypergraph where the edges do not have direction.)

Many existing databases, like GRAKN.AI or HypergraphDB have implemented their own versions of a directed hypergraph. GRAKN.AI maps directed hypergraphs to directed graphs. This schema is easy to translate over to database theory, where directed hypergraphs are often compared to ER diagrams. Entities in ER diagrams are represented as vertices and the relationships as hyperedges [4]. We will proceed with an example.



Hypergraph as a directed hypergraph in SQL

Figure 3

Suppose we want to categorize people who are a member of a certain family and people who are not a member of that family and their pets. Then we would have the relationships *in_family*, *not_in_family* and *pets*. In this example, each member is represented as an entity and we have the relationships of interest as belongs_to, knows, likes and dislikes (Table 1, 2). These relationships can be represented as a directed graph (Figure 2) [4].

Table 1: SQL Table for *pets* Relation

ID	belongs_to
pet_1	person_1
pet_2	person_a

Table 2: SQL Table for *in_family* Relation

ID	knows	likes	dislikes
person_1	person_a, person_b	person_b	
person_2	person_b		person_b

Clearly, we can add more information about these people and their pets, but for tutorial purposes we will stick this. From here, we can find shortest path, betweenness and other properties about the network.

We can model a metabolic or synthetic lethal reaction this way. Both of which only make sense in combination with each other. Further, in synthetic lethal interactions and metabolic pathways, we need to consider the reactants and products as a directed relationship (i.e. $A+B \rightarrow C+D$, where A,B are the reactants and C,D are the products) [3].

Sometimes, however, it makes more sense to model biological processes as an undirected hypergraph. Consider, again, protein-protein interaction (PPI) and gene co-expression networks. These databases are intended to predict function by grouping entities together without direction. In fact, there are several databases in existence that represent these networks as undirected graphs.

Common PPI networks that use undirected graphs are BioGRID, MINT, BIND, DIP, IntAct and HPRD [5].

There aren't many undirected hypergraph technologies in existence. However, HyperGene and HyperPrior offer integrated PPI and gene expression hypergraph models [2] [6]. While HyperGene and HyperPrior integrate genomic expression data, chromosomal spatial information with arrayCGH data to quantify amplification or deletion of large DNA segments on a chromosome, it is unclear how a framework like this can be combined with other features that are important to biomarker discovery.

Hypergraphs have been explored in the context of knowledge representation and graph- object-oriented databases, and database schemas have (sometimes inadvertently) used a hypergraph-based framework. However, there isn't a commonly used hypergraph biological database, likely because biological networks are complex. Fusing that data together in a way that preserves biological meaning from standalone datasets can seem even more challenging. For that reason, Systems Imagination designs our hypergraph database strategically, and we enable our customers to find answers faster using our hypergraph databases.

References

[1] Bretto, Alain. Hypergraph Theory an Introduction. Springer, 2013.

[2] Hwang, TaeHyun, Ze Tian, Rui Kuang, and Jean Pierre Kocher. "Learning on Weighted Hypergraphs to Integrate Protein Interactions and Gene Expressions for Cancer Outcome Prediction." — Experts@Minnesota. Eighth IEEE International Conference on Data Mining, 28 June 2008. Web. 17 May 2017.

[3] Klamt, Steffen, Utz-Uwe Haus, and Fabian Theis. ``Hypergraphs and Cellular Networks." PLoS Computational Biology 5.5 (2009): n. pag. Web.

[4] Klarman, Szymon. ``Modelling Data with Hypergraphs – GRAKN.AI." GRAKN.AI. GRAKN.AI, 19 Apr. 2017. Web. 07 July 2017.

[5] Lehne, B.,and T.Schlitt. "Protein-Protein Interaction Databases: Keeping up with Growing Interactomes." Human Genomics, 3 Apr. 2009. In: Human Genomics, Vol. 3, No. 3, 04.2009, p. 291 - 297.

[6] Tian, Ze, Taehyun Hwang, and Rui Kuang. "A Hypergraph-based Learning Algorithm for Classifying Gene Expression and ArrayCGH Data with Prior Knowledge." Bioinformatics. Oxford University Press, 24 June 2009. Web. 17 May 2017.

Follow us:

@systemsimagin

linkedin.com/company/systems-imagination-inc

Systems Imagination

SYSTEMS IMAGINATION, INC
7730 E Greenway Rd., Ste 205
Scottsdale AZ 85260

480.422.2731
info@siiventures.com
www.systemsimagination.com