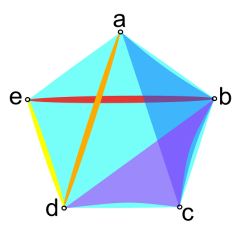
Discovering Higher-Order Interactions in Hypergraphs via Projections: Concepts & Algorithms

Keywords: Hypergraphs, Higher-Order Interactions, Reconstruction, Higher-Order Networks, Algorithms

Extended Abstract

Consider a hypergraph G with unbounded edge size; how much can we learn about the higher-order interactions in G if we only see k-way connections (for each fixed value of $k = 2, 3, 4, \dots$)? In particular, if the $(k+1)$-way interactions are not completely determined by their k-way projections (together with all hyperedges of size $\leq k$), then we would say there is a new higher-order interaction at level $k+1$. As far as we are aware, this represents a new and previously unstudied approach to defining and understanding higher-order interactions in hypergraphs.[1] We develop both the concepts and efficient algorithms for finding such higher-order interactions in real data sets.



a	b	c	d	e
0	1	0	0	1
1	0	0	1	0
0	0	0	1	1
1	1	1	1	1
1	1	1	0	0
0	1	1	1	0

Figure 1: **Example Hypergraph** and its incidence matrix.

To be more precise, consider the $m \times n$ edge-vertex incidence matrix of G. For each fixed set of k out of the n columns (=vertices), we get a set of length-k substrings (hyperedge fragments), called a k-way projection. This can be considered as a decomposition of a given hyperedge into k-bounded component edges. Can all the k-way projections be used to reassemble the original data set? How does the hypergraph that is reconstructed from the k-way projections change as we change the value of k? In this framework, we know which columns (=vertices) we are viewing, but we only see the *set* of those length-k strings (hyperedge fragments). In particular, we do not know which hyperedge fragment came from which original hyperedge(s).

This question was inspired largely by real-world data sets that are naturally hypergraphs or sets of strings. For example, coauthorship hypernetworks (vertices=authors, hyperedges=papers), patent hypernetworks (vertices=technology codes, hyperedges=patents), epidemiological hypernetworks of multiple kinds, and most survey data. For example, a set of survey data where participants were asked a series of 20 yes/no/maybe questions can be modeled as a hypergraph with 60 nodes, one for each question-answer pair, and then each participant's answers give a single hyperedge of size 20. An example from epidemiology is modeling disease spread in a dorm, where each week corresponds to a hyperedge specifying which students were ill that week.

This broad question on data reconstruction opens up into several different interpretations that are interesting to consider. We generally find working with our edge list as a set of strings to be the most clear, and so we will use the string definition below. The most apparent question, and the one that we primarily focus on in this work, is determining the value of k that allows us to perfectly reconstruct our original data set. We give this value a name:

Definition: Given $S \subseteq \{0,1\}^n$, $x \in \{0,1\}^n$ is *k-reconstructible* from S if for every $I \subseteq [n]$ with $|I| = k$, $\exists s \in S$ such that $s|_I = x|_I$. The *k-reconstruction* of S is the set of all strings k-

[1]Our work has recently been published in a conference; details omitted for anonymity.

reconstructible from S, denoted $Recon_k(S)$. A set S is called *k-reconstructible* if $S = Recon_k(S)$. The *point of perfect reconstruction* is the least integer k such that $S = Recon_k(S)$.

Although most algorithmic problems related to k-reconstruction of strings of length n (hypergraphs with n vertices) have obvious $2^{\tilde{O}(n)}$-time algorithms, real-world data sets are often very sparse, in the sense of having many fewer than 2^n hyperedges. In general we prove that finding the point of perfect reconstruction is NP-hard, and we conjecture it is complete for $\Pi_2\mathsf{P}[k,*]$ [1]. Nonetheless, because most real-world data sets are sparse, we seek algorithms that are significantly more efficient in practice.

Our main contribution, in addition to the statement and definition of concepts and questions, is an efficient algorithm for computing the k-reconstruction. Our algorithm involves the creation and analysis of an "overlap graph," similar to the overlap graphs used in DNA reconstruction. To illustrate the idea, we use Fig. 1, and arbitrarily choose the vertex ordering a,b,c,d,e and $k=3$, i.e. we are seeking to find $Recon_3(S)$. The overlap graph (Fig. 2) has one level for each contiguous set of k indices in our ordering: $[a,b,c],[b,c,d],[c,d,e]$. The nodes in each level correspond to the projections of the strings in S onto each size-k index set. There is a directed edge between nodes at one level and the next if they agree on the overlapping $k-1$ indices. Reconstructed strings will then correspond to paths through this graph. In Fig. 2 we highlight the paths corresponding to the original hyperedges from Fig. 1 with the corresponding colors. However, the definition $Recon_k(S)$ involves not only the consecutive k-windows but the non-consecutive ones as well. We show that one can test whether a given string is in $Recon_k(S)$ using the Hitting Set problem which is NP-complete; any paths that are reconstructed in the overlap graph are then checked against all windows using a Hitting Set algorithm. For example, the edge from the last node in the first row to the last node in the second row turns out to not match any string in $Recon_3(S)$, and the corresponding path will get eliminated by an appropriate call to Hitting Set. Starting with the overlap graph significantly reduces the number of calls needed to Hitting Set.

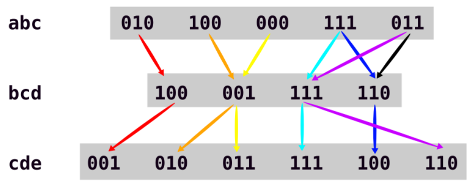


Figure 2: **Overlap graph** for the $k=3$ reconstruction of the data set shown in Fig. 1.

We ran extensive tests on our algorithm's performance. In practice we could easily work with data sets of up to 50 nodes and 200 hyperedges on a low-powered laptop in under 30 seconds. Although this is not near the scale of modern data sets with thousands or millions of nodes, we believe higher-order interactions in such data sets could be elucidated by looking at 50-node subsets at a time. In ongoing work we are continuing to improve the algorithm's efficiency and exploring the combinatorial possibilities for higher-order interactions.

Ethical considerations. We do not believe this problem or its algorithm have any inherent ethical impacts, however as with any tool there must be some care in how our algorithm and its outputs are ultimately applied.

References

[1] Ronald de Haan and Stefan Szeider. "A Compendium of Parameterized Problems at Higher Levels of the Polynomial Hierarchy". In: *Algorithms* 12.9 (2019).