


Fast Algorithms for Finding the Results of Union and Intersection Operations for Bases of Extreme N-Homogeneous Hypergraphs

Beretskiy Igor Sergeevich ¹

*Moscow Aviation Institute (National Research University), 125993, Moscow, Russia
godpingwiner@gmail.com*

Keywords: Hypergraph, Logic algebra, Base, Simplex, Complex, Extremal hypergraph, Perfect hypergraph, K-Uniform hypergraph, Venn-Euler Diagram.

Abstract: This paper is devoted to the study of operations on certain characteristics of extremal k-homogeneous hypergraphs – bases. Usually, operations on hypergraphs are performed via their adjacency matrices, which, with a large dimension, leads to an increasing power complexity of algorithms and requires the disclosure of a large-order adjacency matrix in memory. The paper suggests an alternative approach: the use of union and intersection operations directly over the bases-characteristics of extreme hypergraphs, which potentially leads to an advantage in the speed of the algorithm and unambiguously in the memory used. In general, we present algorithms for quickly finding the results of combining and intersecting bases of extremal n-homogeneous hypergraphs.

1 INTRODUCTION

Graph theory is a dynamically developing field of research that is increasingly being integrated into various areas of human life. Hypergraphs (Egorova, Mokryakov, Vang, 2018) are one of the most important areas of research and are used in various fields of science: from the representation of algebraic structures (Cvetov, 2019; Goltsova, Egorova, Mokryakov, Tsurkov, 2021; Mokryakov, 2011) to the generation of various network structures (Golovkin, 2014) and encryption (Egorova, Mokryakov, Suvorova, Tsurkov, 2021). We should also mention methods for representing a special class of homogeneous hypergraphs – vector. Related to them are algorithms for reconstructing hypergraphs from vertex degree vectors (Mironov, Mokryakov, Sokolov, 2007; Bereckij, Irbitskij, Egorova, Mokryakov, 2020; Mokryakov, Tsurkov, 2011; Kostyanoi, Mokryakov, Tsurkov, 2014.; Bereckij, Irbitskij, Egorova, Mokryakov, Chernova, 2020).

In this paper, we consider perfect and extremal hypergraphs (Mokryakov, 2011), which have special properties. One of them is an additional form of description, the base (Mokryakov, 2011).

A database is a more concise and understandable form of writing an extreme hypergraph. Since only one database uniquely corresponds to one hypergraph, you can work with them without any risk of misinterpretation of data.

When interacting with hypergraphs, it is often necessary to perform some arithmetic operations with them (Bereckij, Irbitskij, Egorova, Mokryakov, Chernova, 2020; Bereckij, 2020; Bereckij, Irbitskij, Egorova, Mokryakov, 2020).

Usually, the above operations are performed via adjacency matrices. However, the use of databases makes it possible not only to write hypergraphs in a simplified form, but also to optimize various arithmetic operations involving hypergraphs (Egorova, Yesenkov, Mokryakov, 2020).

This article discusses algorithms for quickly finding the results of operations for combining and intersecting bases of n-homogeneous hypergraphs.

In the future, it is planned to develop a library based on them, which can be used in a software package for generating encryption keys for a topological algorithm (Egorova, Mokryakov, Suvorova, Tsurkov, 2021) or for analyzing economic models (Kostikov, Mokryakov, Romanenkov, 2020; Kostikov, Mokryakov, Romanenkov, 2019).

¹  <https://orcid.org/0009-0007-3780-102X>

2 BASIC CONCEPTS OF GRAPH THEORY

First, you need to define the basic concepts. A *graph* is an abstract mathematical object that represents a set of graph vertices and a set of edges, that is, connections between pairs of vertices. However, this paper considers *hypergraphs* – a generalization of a graph in which each edge can connect not only two vertices, but also any subset of vertices.

To denote the number of vertices that connect an edge of a graph, the concept of *uniformity* is introduced. Uniformity means that the hypergraphs under consideration have the same dimension of all edges. For example, a graph without loops is a 2-uniform hypergraph with each edge connecting 2 vertices. In a 3-uniform hypergraph, each edge connects 3 vertices, and so on.

Perfection occurs when there is no isomorphic hypergraph with the same degree vector as the graph under consideration. *Extremality* occurs when perfection is satisfied and the degree vector is ordered in non-ascending order (Mokryakov, 201).

To determine the perfection and extremality of hypergraphs, we will consider their vectors using the Hakimi algorithm. To do this, we will first introduce a few concepts.

Let the vector A of \mathbb{Z}_+^n be a vector realized in a graph and $\{G(A)\}$ be the set of its realizations.

A vector A of \mathbb{Z}_+^n , where $n \geq 2$, is called *perfect* if $|\{G(A)\}|=1$. In this case, the only implementation of $G(A)$ is called a *perfect graph*.

A vector A of \mathbb{Z}_+^n , where $n \geq 2$, is called *extreme* if A is a perfect vector. In this case, the only implementation of $G(A)$ is called an *extremal graph*.

A vector $A = (a_{a1}, \dots, a_{an}) \in \mathbb{Z}_+^n$, where $n \geq 2$, is called *strictly reducible* if

$$a_1 = n - 1 - l_A(0),$$

where $l_A(0) = |\{a_i: a_i = 0, 2 \leq i \leq n\}|$

Hakimi's algorithm:

A vector A of \mathbb{Z}_+^n , where $n \geq 2$, is extremal if and only if A is a strictly reducible vector and every reduction vector $A^{(k)}$ constructed from A in the k th reduction step at $1 \leq k \leq n-2$ is a strictly reducible vector.

Every extremal hypergraph has a *base*. Before defining this concept, we introduce several other notations.

A *simplex* is the simplest partitioning figure. Those geometric shapes that can be properly partitioned into simplices are called polyhedral, and

the scheme of partitioning into simplices itself is called a *complex* or (in general) a hypergraph.

For a set of vertices $U(n)$, where $n \geq 2$, and $k \in \mathbb{Z}$, $0 \leq k \leq n-1$, by

$$S^{k+1}(n) = \{\{u_{i_1}, \dots, u_{i_{k+1}}\}: u_{i_1} \in U(n), u_{i_p} \neq u_{i_q} \text{ and } p \neq q\}$$

we denote the set of all $(k+1)$ -element subsets of $U(n)$. A pair of sets $\{U(n), S^{k+1}\}$, where $S^{k+1} \subseteq S^{k+1}$, which we denote by

$$G^k = G^k(U(n), S^{k+1} = S^{k+1}(G^k)),$$

it is called a *k-complex* (Mironov, Mokryakov, Sokolov, 2007).

Consider the set of all cubic symmetric binary matrices that are adjacency matrices of n -vertex 2-complexes $\{X_n^{(3)} = (x_{ijk}): 1 \leq i, j, k \leq n\}$.

On the set of all elements consisting of three pairwise distinct indices

$$I_n^3 = \{\{i, j, k\}: 1 \leq i < j < k \leq n\},$$

A partial order relation is introduced that defines a partial order for 2-dimensional simplices of $S^3(n)$ and any set of simplices of $S_0^3 \subseteq S^3(n)$. Elements of I_n^3 are called *triples*.

On the elements of the set I_n^3 , we introduce a partial order relation: let $\{i, j, k\} \geq \{p, q, l\}$, if $i \geq p, j \geq q, k \geq l$ and $\{i, j, k\} > \{p, q, l\}$ for $\{i, j, k\} \geq \{p, q, l\}$ and $\{i, j, k\} \neq \{p, q, l\}$.

Let G^2 be an arbitrary n -vertex 2-complex with adjacency matrix $X_n^{(3)}(G^2) = (x_{ijk})$. We will use the notation $G^2 \cong (x_{ijk}) = X_n^{(3)}$. For the set of triples corresponding to simplices of the complex G^2 , we introduce the notation

$$I_n^3(G^2) = \{\{i, j, k\} \in I_n^3: x_{ijk} = 1\}.$$

After entering all these terms, we will move on to the very concept of a database.

Let $G^2 = G^2(U(n), S^3(G^2)) \cong (x_{ijk})$ is an arbitrary 2-complex, where $S^3(G^2) \neq \emptyset$. The set $\bar{I}_n^3(G^2) = \{\{i, j, k\}\}$ is called *the base* for the complex G^2 if the following conditions hold:

- If $\{p, q, l\} \in \bar{I}_n^3(G^2)$, then $\{p, q, l\} \in I_n^3(G^2)$, i.e. $\bar{I}_n^3(G^2) \in I_n^3(G^2)$;
- If $\{p_1, q_{p1, q1}, l_{11}\}, \{p_2, q_{q2}, l_{12}\} \in \bar{I}_n^3(G^2)$, then the partial order relation for these triples is not defined.

c) For $\{i, j, k\} \in I_n^3(G^2) \setminus \bar{I}_n^3(G^2)$, there exists a triple $\{p, q, l\} \in \bar{I}_n^3(G^2)$ such that $\{i, j, k\} < \{p, q, l\}$;

d) If $\{i, j, k\} < \{p, q, l\}$, where $\{p, q, l\} \in \bar{I}_n^3(G^2)$, then $\{i, j, k\} \in I_n^3(G^2)$.

After entering the basic concepts, let's look at examples of how they are implemented in 2-homogeneous hypergraphs.

3 OPERATIONS ON HYPERGRAPHS

Hypergraphs are inherently sets, so you can legally perform binary operations on them. However, the question arises whether the properties of perfection and extremality are still preserved. Will the result have a base? If so, then it will be possible to perform operations on databases directly, since the result will also be the base.

Consider 2-homogeneous extremal hypergraphs G_3 and G_4 .

G_3 :

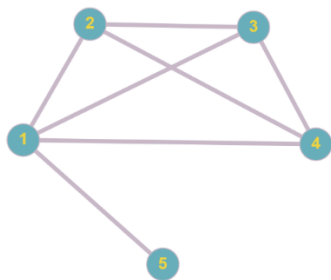


Figure 2.1: Hypergraph G_3 .

The degree vector is $(4,3,3,3,1)$. Base— $\{\{3,4\}, \{1,5\}\}$.

G_4 :

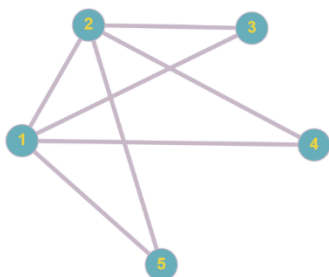


Figure 2.2: Hypergraph G_4 .

The degree vector is $(4,4,2,2,2)$. The base is $\{2,5\}$.

Results of operations:

Unification.

$$G_3 \cup G_4 = \{\{3,4\}, \{1,5\}\} \cup \{2,5\} = \{\{3,4\}, \{2,5\}\}$$

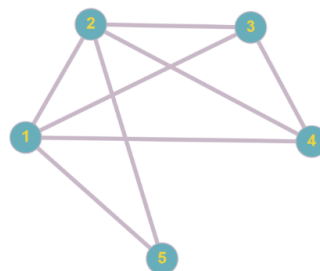


Figure 2. Ошибка! Закладка не определена.: Result of the merge operation.

The vector of the resulting graph is $(4,4,3,3,2)$.

Let's check it for perfection:

$$A \quad 4 \ 4 \ 3 \ 3 \ 2$$

$$A' \quad 3 \ 2 \ 2 \ 1$$

$$A'' \quad 1 \ 1 \ 0$$

The vector is perfect, which means that the resulting graph is extreme.

1) Intersection.

$$G_3 \cap G_4 = \{\{3,4\}, \{1,5\}\} \cap \{2,5\} = \{\{1,5\}\}$$

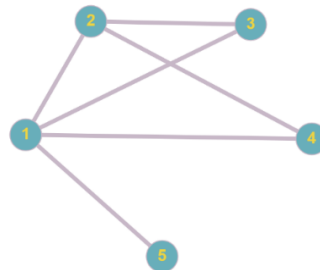


Figure 2.4: Result of the intersection operation.

The vector of the resulting graph is $(4,3,2,2,1)$.

Let's check it for perfection:

$$A \quad 4 \ 3 \ 2 \ 2 \ 1$$

$$A' \quad 2 \ 1 \ 1 \ 0$$

The vector is perfect, which means that the resulting graph is extreme.

2) XOR.

$$G_3 \oplus G_4 = \{\{3,4\}, \{1,5\}\} \oplus \{2,5\} = \{\{2,5\}, \{3,4\}\}$$

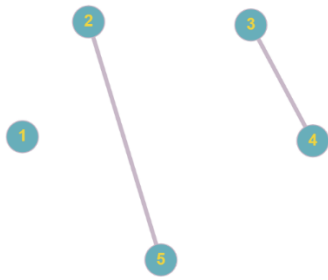


Figure 2.5: Result of the XOR operation.

The vector of the resulting graph is (0,1,1,1,1). Obviously, the result is perfect, but not extreme. Accordingly, the resulting graph is perfect, but not extreme.

3) Equivalence.

$$G_3 \equiv G_4 = \{\{3,4\}, \{1,5\}\} \equiv \{2,5\}$$

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

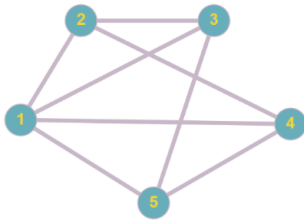


Figure 2.6: Result of the equivalence operation.

The vector of the resulting graph is (4,3,3,3,3). Let's check it for perfection:

$$A \quad 4 \ 3 \ 3 \ 3 \ 3$$

$$A' \quad 2 \ 2 \ 2 \ 2$$

$$A'' \quad 1 \ 1 \ 2$$

At the 2nd step, we got a non-perfect vector, which means that the desired one is not one. Accordingly, the resulting graph is a general graph.

4) Schaeffer's stroke.

$$G_3 | G_4 = \{\{3,4\}, \{1,5\}\} | \{2,5\}$$

$$= \{\{2,5\}, \{3,4\}, \{3,5\}, \{4,5\}\}$$

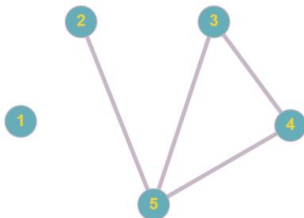


Figure 2.7: Result of the Schaeffer's Bar operation.

The vector of the resulting graph is (0,1,2,2,3). Let's check it for perfection:

$$A \quad 3 \ 2 \ 2 \ 1 \ 0$$

$$A' \quad 1 \ 1 \ 0 \ 0$$

The vector is perfect, which means that the resulting graph is perfect, but not extreme.

As a result, only the union and intersection operations preserve the extreme and perfect properties, and in the future, you can use these operations directly to work with databases. XOR, equivalence, and Schaeffer's stroke do not preserve both extreme and perfect properties, so you can't use them directly for working with databases.

3 INTERSECTION AND UNION ALGORITHMS FOR HYPERGRAPH BASES

Currently, there is no standard algorithm for quickly finding the results of binary operations on extreme hypergraphs. Current algorithms require the construction of adjacency matrices, which adds complexity to calculations. This article describes algorithms that allow you to get the same result without constructing and using bulky matrices.

3.1 Description of the Merge Algorithm

Let's say you need to combine two hypergraphs (1) (2) and (3) (4), where (1) (2) (3) (4) these are their bases. First, we write out all these bases in one set and sort them in ascending order.

Then we start going from the maximum element to the minimum, comparing the current element and the previous one.

Then there are 2 cases - either element n-1 is included in element n or not. If it is included, we skip it, and compare n and n-2 already. If it is not included, then element n goes to the resulting database, and we start comparing n-1 with n-2. This way we go through the entire array.

This algorithm does not change in any way from the dimension of hypergraphs and the number of bases, only the number of comparison iterations increases/decreases.

3.2 Merge Algorithm

1. We'll put all the databases in one array, number them, and sort them. (n1, n2, ... ni).
2. Let's compare the element ni and ni-1.

3. If the element n_{i-1} is included in the element n , then the counter i is reduced and we return to point 2.
4. If the element n_{i-1} is not included in the element n , then the element n goes to the final result base, then the counter i is reduced and we return to point 2.
5. If we reach the beginning of the array, then the last remaining element goes to the final result base.

3.3 Description of the Intersection Algorithm

The intersection algorithm differs slightly from the union algorithm due to the fact that bases that were not originally present may appear during the intersection.

Let's say we need to find the result of the intersection of two hypergraphs (1) (2) and (3) (4), where (1) (2) (3) (4) these are the bases. First, we write out all these bases in one set and sort them in ascending order. In this case, the bases of the first hypergraph will be assigned the flag "1", and the bases of the second hypergraph will be assigned the flag "2".

Then we go from the first element to the last and see if the elements i and $i+1$ belonged to the same hypergraph. There are 2 cases here.

1) They belonged to different hypergraphs.

In this case, we find their intersection and write it to the resulting array.

2) They belong to the same hypergraph

Then we start looking at elements $i+2$, $i+3$, and so on until we come across a base from another hypergraph. As soon as we stumble, we find the intersection of these two elements, after which we replace element i with the result of the intersection, change its "belonging" to the opposite hypergraph, and move on to point 1, since now they belong to different hypergraphs with the following elements.

After we go through the entire array in this way, we need to apply the union algorithm to the resulting hypergraph, since the base of the hypergraph is the minimum set of elements, but after applying the algorithm above, we could have extra elements that are part of others.

3.4 Intersection Algorithm

1. We'll put all the databases in one array, number them, and sort them. (n_1, n_2, \dots, n_m). At the same time, we will assign flags to the elements that characterize their belonging to a particular hypergraph. Set $i=0$.
2. Compare the element n_i and n_{i+1} .
3. If they belong to different hypergraphs, we find their intersection and write them to the resulting array. Increment the counter i by 1.
4. If they belong to the same hypergraph, compare the elements n_i and n_{i+k} ($k=2, 3, \dots$) until the element n_{i+k} is from another hypergraph. Find their intersection and replace the n_i element with it. Change the hypergraph membership flag for this element to n_{i+k} . Go to step 3.
5. If we reach the end of the array, then the last remaining one is discarded. If we reach the end of the array within step 4, then the n_i element is declared last and we move on to step 6.
6. In the resulting array, we apply the merge algorithm.

4 CONCLUSION

The hypergraph theory still has a wide field of research, which may lead to a significant number of important scientific discoveries in the future. With the development of information technology, it is becoming less and less a purely theoretical discipline, and more and more begins to influence everyday life.

These algorithms are only a starting point in the study of hypergraph bases and operations with them. However, the development of encryption algorithms using this technology has already begun.

REFERENCES

- Egorova, E. K., Mokryakov, A. V., Vang, L., 2018. Development of Hypergraph Theory. *J.~Computer and Systems Sciences International*, V. 57, Issue 1. Pp. 109-114.
- Cvetov, V. P., 2019. Algebrý n-mestnyh otnoshenij. *Sbornik trudov ITNT-2019*. pages 916-923.
- Goltsova, T.Y., Egorova, E. K., Mokryakov, A. V., Tsurkov, V.I., 2021. Signatures of Extremal 2-Uniform Hypergraphs. *J.~Computer and Systems Sciences International*, V. 60, Issue 6. Pp. 904-912.

- Mokryakov, A.V., 2011. Hypergraphs as Algebraic Structures. *J.~Computer and Systems Sciences International*, V. 50, Issue 5. Pp. 734–740.
- Golovkin, YU. B., 2014. Primenenie nechytokih gipergrafov v modelyah generacii WEB-komponentov. *Izvestiya vysshih uchebnyh zavedenij. Priborostroenie*, № 9. pages 47-53
- Egorova, E. K., Mokryakov, A. V., Suvorova, A.A., Tsurkov, V.I., 2021. Algorithm of Multidimensional Data Transmission Using Extremal Uniform Hypergraphs. *J.~Computer and Systems Sciences International*, V. 60, Issue 1. Pp. 69–74.
- Mironov, A.A., Mokryakov, A.V., Sokolov, A.A., 2007. About Realization of Integer Non-Negative Numbers Tuple into 2-Dimensional Complexes. *Applied and Computational Mathematics*, T. 6. № 1. pages 58-68.
- Bereckij, I.S., Irbitskij, I.S., Egorova, E.K., Mokryakov, A.V., 2020. Algoritmy vosstanovleniya k-odnorodnyh gipergrafov po vektoru stepenej svoih vershin. *Sovremennaya nauka: aktual'nye problemy teorii i praktiki. Seriya: Estestvennye i tekhnicheskie nauki*, № 8. pages 31-36.
- Mokryakov, A.V., Tsurkov, V.I., 2011. Reconstructing 2-Complexes by a Nonnegative Integer-Valued Vector. *Automation and Remote Control*, V. 72, Issue 12. Pp. 2541–2552.
- Kostyanoi, D.S., Mokryakov, A.V., Tsurkov, V.I., 2014. Hypergraph Recovery Algorithms from a Given Vector of Vertex Degrees. *J.~Computer and Systems Sciences International*, V. 53, Issue 4. Pp. 511–516.
- Bereckij, I.S., Irbitskij, I.S., Egorova, E.K., Mokryakov, A.V., Chernova, T.A., 2020. Programmnyj kompleks nahozhdeniya rezul'tatov operacij nad ekstremal'nymi gipergrafami. *Sovremennaya nauka: aktual'nye problemy teorii i praktiki. Seriya: Estestvennye i tekhnicheskie nauki*, № 8. pages 37-45.
- Bereckij, I.S., 2020. Zamknutost' operacij nad klassom ekstremal'nyh odnorodnyh gipergrafov. Hmelevskij I. D. – Gagarinskie chteniya – 2020: XLVI Mezhdunarodnaya molodyozhnaya nauchnaya konferenciya: Sbornik tezisov dokladov: M.; Moskovskij aviacionnyj institut (nacional'nyj issledovatel'skij universitet). pages 197
- Bereckij, I.S., Irbitskij, I.S., Egorova, E.K., Mokryakov, A.V., 2020. Operacii nad k-odnorodnymi ekstremal'nymi gipergrafami. *Sovremennaya nauka: aktual'nye problemy teorii i praktiki. Seriya: Estestvennye i tekhnicheskie nauki*, № 9. pages 49-54.
- Egorova, E.K., Yesenkov, A.S., Mokryakov, A.V., 2020. Operations over k-Homogeneous Hypergraphs and Their Vectors of the Degrees of the Vertices. *J.~Computer and Systems Sciences International*, V. 59, Issue 3. Pp. 381–386.
- Kostikov, Y.A., Mokryakov, A.V., Romanenkov, A.M., 2020. Specialized platform for the analysis of mathematical models in economics. *AMAZONIA INVESTIGA*, V. 8, Issue 23. pages 781-794.
- Kostikov, YU.A., Mokryakov, A.V., Romanenkov, A.M., 2019. Cpecializirovannaya platforma analiza ekonomicheskikh modelej. *Ekonomika: vchera, segodnya, zavtra*, T. 9, № 5-1. pages 218-230.